REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
05-05-2009

2. REPORT TYPE

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
Development of an Integrated Robotic Radioisotope Identification and Location System

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Moore, Christina J. (Christina Jean), 1987-

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Naval Academy
Annapolis, MD 21402

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Trident Scholar Project Report no. 380 (2009)

12. DISTRIBUTION / AVAILABILITY STATEMENT
This document has been approved for public release; its distribution is UNLIMITED

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This project involves integrating a commercially available Hyper Pure Germanium (HPGe) system within a robotic base in order to inspect an area for either radioisotopes that could be used for a radiological dispersal device (RDD) or are classified as Special Nuclear Material (SNM). In operation, at a given location in the room, the robot rotates about its circumference searching for radionuclides. The robot then estimates the direct ion of a source of interest. The robot can then be moved to a new location to repeat the process. With three locations and a direction to the course from each, a program then calculates the estimated location of the source of interest. The project also investigated statistical properties related to isotope identification software and the effects of changing the settings of the program and the setup of the experiment.

15. SUBJECT TERMS
Isotope, location, identification, Hyper Pure Germanium, robot

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**		60	**19b. TELEPHONE NUMBER** *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

U.S.N.A. --- Trident Scholar project report; no. 380 (2009)

Development of an Integrated Robotic Radioisotope Identification and Location System

by

Midshipman 1/c Christina J. Moore
United States Naval Academy
Annapolis, Maryland

(signature)

Certification of Adviser(s) Approval

Professor Martin E. Nelson
Mechanical Engineering Department

(signature)

Professor Mark J. Harper
Mechanical Engineering Department

(signature)

Associate Professor Randy Broussard
Weapons and Systems Engineering Department

(signature)

Acceptance for the Trident Scholar Committee

Professor Carl E. Wick
Associate Director of Midshipman Research

(signature)

SUMMARY

The goal of this project is to make an integrated robotic radioisotope identification and location system. This project involves integrating a commercially available Hyper Pure Germanium (HPGe) system with a robotic base in order to inspect an area for either radioisotopes that could be used for a radiological dispersal device (RDD) or are classified as Special Nuclear Material (SNM). In the current security climate, such a machine would be extremely useful.

However, designing and building a 'smart' machine that uses a HPGe detector presented the challenge of finding the right balance between speed and accuracy. The longer the inspection time, the more accurate the search is, but the longer it takes to complete. Ideally, the system should be optimized to balance speed with accuracy. Also, all data from the sensor has statistical fluctuations, which further challenges the system in locating the source.

At a given location in the room, the robot rotates circumferentially searching for radionuclides. The robot then estimates the direction of a source of interest. The robot can then be moved, using inputted instructions, to a new location to repeat the process. With three locations and a direction to the source from each, a separate program then calculates the estimated location of the source of interest.

The project also investigated the statistical properties related to isotope identification software and the effects of changing the settings of the program and the setup of the experiment.

Key Words
Isotope
Location
Identification
Hyper Pure Germanium
Robot

ACKNOWLEDGMENTS

The successful completion of this project would not have been possible without help from many people and organizations.

I would like to thank my Academic Advisors, Professor Nelson, Professor Harper and Assistant Professor Broussard. Without their willingness to indulge me in such an ambitious initial research plan, none of this would have started. Without their constant advice and guidance on the results, none of the conclusions could have been justifiably reached.

Secondly, I would like to thank all of the Naval Academy support staff who helped me, especially Mr. Dellikat. I drew on his patient help and vast practical experience throughout the entire course of the project. I would also like to thank Mr. Bradshaw for his aid and expertise in systems integration, to get the all of the parts to work together in the construction phase. Also I would like to thank the staff at the Machine shop for their initial design of the parts and their manufacture of the raw parts. Finally, I would like to thank the electrical support staff for being available to consult on the power supply when it was sporadically needed.

I would also like to thank Professor Bishop, Associate Professor Feemster, Professor Wick, Professor DeMoyer and Associate Professor Espizito for being available when I needed to consult with an expert in a field.

Before beginning this project, I took part in an internship over the summer, which helped me to prepare for this project. I would like to thank R. Larry Karch for organizing my time at DTRA, John Smalling and Susi Brockman for organizing my time with ORTEC and Mr. Von Sudderth for taking me under his wing at the Department of Energy. I would especially like to thank Mr. Randy Meyers for his creation of the “Data Acquisition Program” and the “IDM Demo Program”.

Finally, and in someways most importantly I would like to thank DTRA for lending me the IDM sensor to construct the robot and United States Naval Academy Trident Scholar Committee for giving me the chance to do this project.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

FIGURES

TABLES

ACRONYMS

AC	Alternating Current
Ci	Curie
DAP	Data Acquisition Program
DTRA	Defense Threat Reduction Agency
DR	Detection Rate
HEU	Highly Enriched Uranium
IED	Improvised Explosive Device
IDM	Interchangeable Detection Module
LIDAR	Laser Range finder
LED	Light Emitting Diode
RC	Radio Control
RDD	Radiological Dispersal Devices
RPM	Revolutions Per Minute
SNM	Special Nuclear Material
Q	Statistical parameter of source identification
UPS	Uninterruptible Power Supply

SECTION 1

BACKGROUND AND STATEMENT OF PROBLEM

In the current security climate, it is essential that terrorist organizations do not obtain nuclear material. A vital part of this is preventing this material being smuggled it into the country. It is also useful to be able to search an area of suspected radiation and/or booby-trapping without risking personnel. An autonomous system capable of this would have much to contribute to both of these scenarios, especially if it could identify the source of the radiation. Even a remote control system would reduce the risk to human personnel, and lay the foundations for an autonomous system to follow.

1.1 THE NEED FOR DETECTION

Nuclear material presents an especially great hazard to public safety if it falls into the hands of terrorists and then is smuggled into the US. The types that could be used in an attack fall into two broad categories: special nuclear material (SNM) and radiological dispersal devices (RDD).

SNM is any fissile material whose properties can be used to either power a nuclear power plant or to create a nuclear weapon. Figure 1-1 shows an image of two main types of SNM. An example of SNM would be highly enriched uranium (HEU), which can be 93% U^{235} and 7% U^{238}.

Figure 1-1. Uranium (left) and plutonium (right).

Another SNM is plutonium, also shown in Figure 1-1, in which Pu^{239} is the most common fissile isotope that can be used in weapons or fuel for nuclear power reactors. It is possible to move SNM around, as shown by the International Atomic Energy Agency's data from 1993 to 2001, which found over 180 incidents of nuclear smuggling in more than 40 countries. (Anzelon, 2001) Though there are treaties that attempt to prevent the spread of nuclear technology, these treaties do not require that the countries prevent their nuclear material from being stolen, even if it could be used to make a nuclear weapon (Bunn 2001). If enough small quantities could be smuggled into our country, it is theoretically possible to make a simple crude atomic bomb by experimentation.

There is also a fear of RDD or 'dirty bombs', where conventional explosives are used to disperse large amounts of radioactive material over a large area, causing radiation contamination to all who come in contact with it. Though this initially sounds devastating, many experts suspect that the RDD would be a more effective weapon of mass disruption than mass destruction. The impact of an attack would depend on such variables as the weather, how fast it is possible to evacuate the local populace, and variables under the RDD makers' control, like the type and quantity of the radioactive material used, the method of dispersal and the placement of the explosive. This type of bomb is much easier to construct than a small nuclear device, as it requires no knowledge of higher-level physics beyond that used by the average Improvised Explosive Device (IED) manufacturer (FEMA, 2006).

The main stumbling block in the building of an RDD would be the acquisition of a large enough amount of suitable material. Though there are many radioactive isotopes used in industry, medicine, and research, only some would be useful as the active component in an RDD. Many materials, like the small sources used in smoke detectors, are sealed and so weak that they would have very little effect, even if thousands of them were collected. A sample set of isotopes that could be used in an RDD, their specific activity, half-life, and the gamma energies that characteristically identify them are given in Table 1-1, taken from (OSHA 2004).

Table 1-1. Isotopes suitable for RDD.

Isotope	Symbol	Specific Activity (Ci/g)	Half Life (years)	Gamma Energies(KeV)
Americium-241	Am^{241}	3.5	430	26.3, 59.4
Cesium-137	Cs^{137}	88	30	662
Cobalt-60	Co^{60}	1100	5.3	1170, 1330
Iridium-192	Ir^{192}	9200	0.2	308, 316, 468
Radium-226	Ra^{226}	1	1600	186

Radioactive waste from nuclear power plants and dismantled weapons sound like tempting targets for a terrorist to use in an RDD, but its usefulness depends on the material's physical form. Additionally, such suitable waste is generally closely guarded, and that which is not closely guarded would be very hard to use in an RDD (Shea 2004).

1.2 GAMMA RAY SPECTROSCOPY TERMINOLOGY AND CONCEPTS

Detecting a radioisotope is all about picking up as many incoming gamma rays as possible and measuring the gamma ray's energy. There are three factors that control the count rate of a radioisotope source:

- Time: the number of emitted gammas that interact with the detector is directly proportional to the count time.
- Distance: the closer the source is to the detector, the more radiation the detector will pick up because the solid angle between the source and detector is larger.
- Shielding: The less shielding present between the source and the detector, the more radiation reaches the detector due to lower attenuation of the incident gamma rays.

The geometry of the source itself will also affect the ease of detection. For the same mass of isotope, a plate facing the detector would give a far higher reading than a sphere. This is because most of the gamma radiation emitted in the center of a SNM source is absorbed within the source. SNM sources can be treated as surface sources and the larger the surface area, the more radiation there is to detect. (Harper 2007)

State-of-the-art high resolution gamma spectroscopy makes it possible to simultaneously detect the most important RDD and SNM isotopes while at the same time reducing the number of false positives. With a traditional Geiger counter, only the presence or absence of an unknown radioisotope can be determined.

By measuring the photopeaks, a gamma spectroscopy system can identify the radioisotopes that are present. A photopeak is a resonance in gamma ray readings, centered on the characteristic photon energy emitted by the gamma source. When an isotope decays and its nucleus goes from one energy state to another, a gamma ray is released. The difference between these two states is characteristic of the specific radioisotope. A large body of research has gone into identifying which gamma ray energies are associated with each radioisotope. This means that gamma spectroscopy systems can perform isotope identification. Resolution in gamma spectroscopy is a measure of the sharpness of the photopeak and is defined as the full width half maximum expressed in energy divided by the gamma energy of the source material. (Knoll, 2000)

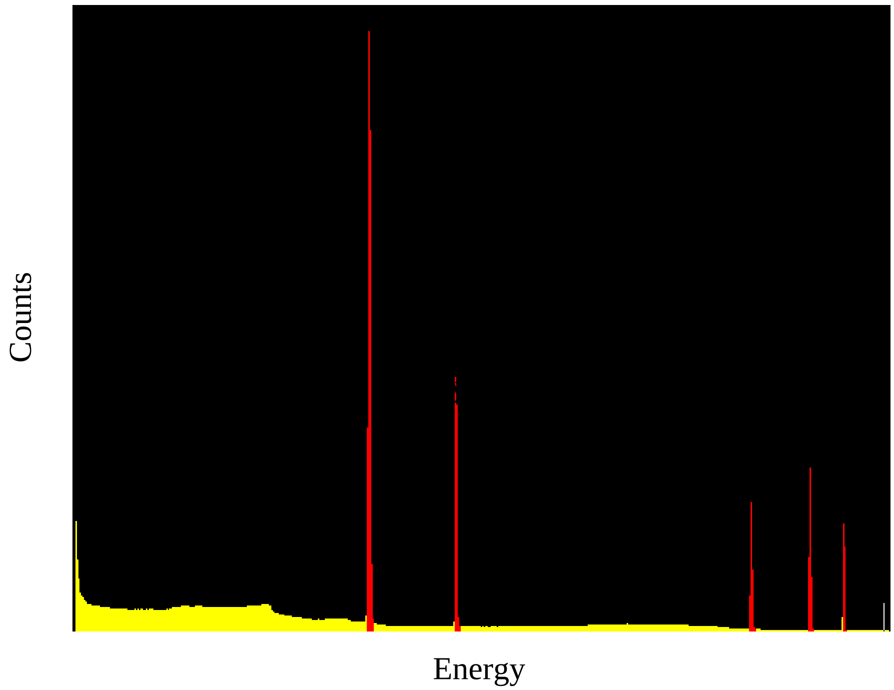

Figure 1-2. HPGe Peak Detection Gamma Ray Spectrum (Delikat 2009)

Some decay modes are dramatically more probable than others, which causes the major peaks. Other decay modes only happen rarely or are actually the emissions of radioisotope daughter products, and these produce the minor peaks. Figure 1-2 is a gamma spectrum obtained with a NaI gamma spectroscopy system for Cs^{137} my Mr. Delikat in the Nucleonics Lab. Cs^{137} is a potential isotope that could be used in an RDD. The red portion shows a Cs^{137} photopeak, while the yellow represents background, Compton scattering, and bremsstrahlung radiation

The photopeaks associated with different elements are extremely well established. For example HEU, which is predominately U^{235}, can be identified by the high intensity 186 KeV gamma photopeak U^{235} creates. Weapons grade plutonium (93% Pu^{239}) can be identified by either its 375 KeV or the 414 KeV gammas. Even when these two isotopes decay, many of their decay products are also strong gamma emitters, which can possibly be used to determine the presence or absence of SNM (Nelson 2006).

Another issue with detection is the efficiency of the detector. The efficiency measures the fraction of the radiation that passes through the detector that appears in the photopeak. The higher the efficiency, the more counts the detector picks up and the less time it has to take gathering data before it can reach a statistically valid conclusion about the presence of a particular isotope. Larger detectors tend to be more efficient, but the materials used are also a factor.

1.3 SUMMARY OF THE ORTEC IDM HPGE DETECTOR

For this project, a detection system that employs a high purity germanium detector (HPGe) was used. This detector has been chosen over another option (sodium iodide (NaI)) because NaI has a resolution of 6-7% of the incident energy, while HPGe has a resolution of 0.1% when counting Cs^{137}, and this leads to more accurate results. It is entirely possible to find two isotopes with photopeaks that occur extremely close to each other with a HPGe detector, but not with a NaI detector, as Figure 1-3 shows. The NaI Detector is shown on the left, and the HPGe is shown on the right. The better the detector's resolution, the easier it is to distinguish the presence of different radioisotopes.

Figure 1-3. Comparison of NaI (left) and HPGe (right) detectors (ORTEC 2009).

HPGe is a semiconductor solid state detector operated with an applied electric field of several thousand volts. The electrons in the crystal stay in their valence band until hit by the gamma radiation, causing the electrons to move from the valence to the conduction band. These electrons are then collected, amplified, and counted, giving a spectrum.
The energy scale of the detector is determined by performing a calibration using sources with known gamma energies. Background radiation, which causes increased noise in the signal, can be accounted for by sophisticated algorithms within the spectroscopy system.

Figure 1-4. ORTEC Interchangeable Detector Module (IDM).

The project will use the HPGe Interchangeable Detector Module (IDM) (shown in Figure 1-4) by ORTEC as the detector (ORTEC, 2009). This instrument is a very sophisticated sensor and has a built in USB2 connection which allows it to communicate with a laptop computer. The IDM itself is 47.3 cm long, 45.6 cm wide, 22.9 cm high and weighs 27.2 kg, (50 lbs) which made mounting it challenging. The IDM must be cooled for 4-6 hours before it can detect radiation. This cool down is accomplished with a small sterling engine.

Using proprietary software that can identify 71 isotopes and materials shown in Appendix A, the IDM can be used to identify which sources are present from the gamma peaks and calculate the Q value associated with the identification. This calculation generates a Q value associated with every isotope in the software's library. The Q value or 'quality factor' is defined in equation 1.1 by (Keyser 2008).

$$Q \equiv \frac{G - B}{\sigma_N} \tag{1.1}$$

where

G = gross counts
B = background counts
σ_N = uncertainty in the net peak area

If the Q value is greater than or equal to 5, the IDM software reports a strong confidence that the isotope is present. If in fact that isotope is not present, then the software would have created a false positive. This can occur when multiple isotopes are present, which have close gamma photopeaks as shown in Figure 1-3. If the Q value is less than or equal to 2.15, then the software assumes that isotope is not present and does not report the result. If in fact the isotope is present, then the system has created 'false negatives'. In this study, the detection rate is used to mean that the system finds the isotope with at least some confidence when in fact it is present and which will be discussed in more detail later in the paper.

1.4 OBJECTIVES

The project aim was to create an autonomous platform that could be used to search and locate radioactive material without human assistance. Though this was not completely achieved, major steps in that direction were taken. The project did design, construct and test an integrated robotic system to locate and identify radioactive isotopes. The method of approach used in this project consisted of the following seven key steps listed below.

- Design and build a mobile chassis
- Mount detector (ORTEC IDM) on chassis
- Design and install motion control system

- Integrate commercial radioisotope identification software with motion control and location systems in MATLAB
- Investigate system response to radioisotopes
- Integrate IDM response with robot bearing
- Optimize settings to minimize search time

The first four points are discussed in detail in Section 2. Appendix B contains the specific operating instructions, while Appendix C gives a troubleshooting and further development guide. The results obtained in using the system are given in Section 3 and cover the last three points.
.

SECTION 2

ROBOTIC CHASSIS CONSTRUCTION

The first phase of this project was the construction of the actual robotic chassis. This consisted of designing a base, ordering the appropriate parts, assembling the robot, troubleshooting the power supply and creating precision control of the robot's motion. Many lessons were learned along the way and are discussed in more detail in section 4.3.

2.1 PHYSICAL REQUIREMENTS, DESIGN, AND CONSTRUCTION

The most basic requirements of the robot are that it be able to supply continuous uninterrupted Alternating Current (AC) power to the sensor and support the sensor's 50 lb weight. The AC power must be able to supply up to 300 watts while the sensor is cooling and 70-90 watts once the system is cooled. It must also be able to tilt the sensor, for three dimensional (3D) searching in future projects, turn on the spot to aid searching (i.e. zero turn radius), and it should have a low maximum speed to protect the sensor. No robotic bases existed at the USNA that met all of these requirements, so a new system had to be designed and built.

2.1.1 Design

The basic arrangement is shown in Figure 2-1 below. To aid in the creation of the zero turning radius, the decision was made to have the robot chassis be circular, with different components on different layers. The robot uses the ES308 Rabbit board for motor control. The ES308 Rabbit board was developed for a Systems engineering class (ES308) taught here in the Academy. It uses one master Rabbit chip (i.e. processor) with many protective and slave chips surrounding it. It was designed for this type of application and has, among other things, purpose built motor control outputs. Motion control will be discussed in section 2.3. For position calculations, the robot uses a LIDAR (laser range finder), detailed in section 2.4. For power, the robot uses an Uninterruptible Power Supply (UPS), which will be detailed in section 2.2.

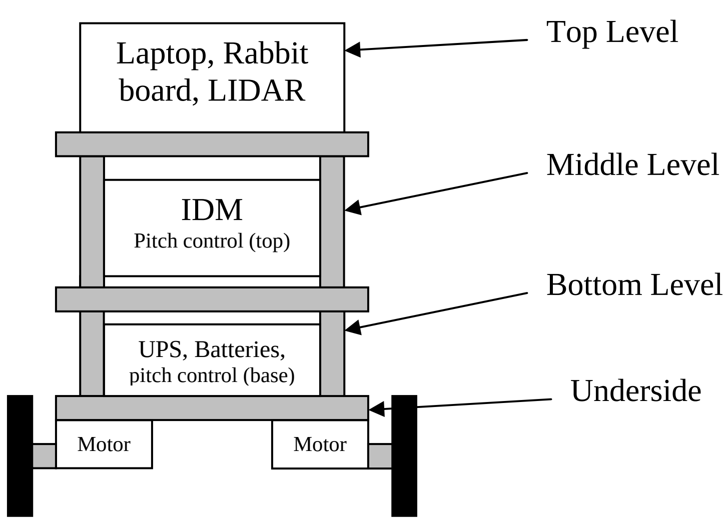

Figure 2-1. Basic layout of robot.

To pick the specific parts, some preliminary estimates of power requirements of the system and the total system weight were calculated. The numbers used to estimate the power requirements are shown in Table 2-1

Table 2-1 Estimation of total robot power

Item	Part Chosen	Wattage used (watts)
Detector	IDM	50
LIDAR	SICK LMS-200	20
Motor	Brevel 420-652 12V with encoder	5
Laptop	HP 8510W - Notebook	45
	Total	120

As shown in Table 2-1, preliminary estimations calculated that the whole system needed 120 watts of AC power for normal operation. Consequently, a battery power of 150 watts was selected, allowing 30 watts for safety margin purposes. Assuming a run time of 4 hours when the cooled down detector is operating on wall power and a current draw of 12.5 amperes, the battery life of the robot is of 50 amp-hours is required. The batteries selected can supply 52 amp-hours. Half inch aluminum was determined to be the best structural material to handle the load and minimize system weight.   The numbers used to estimate the total system weight are shown in Table 2-2.

Table 2-2 Estimation of total weight of robot.

Item	Part Chosen	Weight (lb)
Detector	IDM	50.0
UPS	APC Spart-UPS 750VA 120V	14.3
Inverter	PI-800W Whistler Continuous Power Inverters	15.0
Batteries	2 X UB12260 Sealed Lead Acid (AGM)	41.0
Battery tenders	Battery Tender Junior 12 Volt 021-0123	4.0
LIDAR	SICK LSM-200	2.0
L. Batteries	UB1222 Sealed Lead Acid (AGM)	4.1
Chassis	Aluminum Plates (30” Diameter X 0.5” inch thick)	125.2
Motor	Brevel 420-652 12V with encoder attached	12.0
Laptop	HP 8510W - Notebook	6.1
	Total	273.8

The total of 274 lbs was rounded up to 300 lbs for the purposes of calculating the needed torque. Torque was calculated to accelerate the whole system to a maximum speed in 2 seconds up a wheel chair ramp (5^o of incline) with 8 inch diameter wheels, which led to a required torque of 112 inch lbs. The calculation of torque is shown in Table 2-3.

Table 2-3 Estimation of required torque for robot.

Parameter	Value	
Weight	300.0	lb
angle of incline	5.0	deg
force of gravity	26.1	lb
Mass	9.3	slugs
Wheel diameter	8.0	in
Max RPM	12.0	RPM
Max Speed	0.4	ft/sec
Time	2.0	sec
Acceleration	0.2	ft/sec^2
Force of accleration	1.9	lb
Total force	28.1	lb
Torque	112.4	in lb

The robot uses a matched pair of Brevel 420-652 12V motors with built in encoders. Experience showed that they draw very little current, move at a very low speed but have an extremely high torque. Though they seemed ideal, it was not possible to locate a specifications sheet to confirm this. Assoc. Prof. M. Feemster is an expert on controlling motors from within USNA's Systems engineering department. He was consulted to help calculate the maximum motor torque. However, the experimental set up was unable to find the upper limits of the motor. Fortunately, the upper limits of the test were far higher than the calculated needs of this application, so it was felt that these motors would more than suffice.

The subsystems can be broken down as follows:

- Sensors
  - Nuclear Detector: ORTEC HPGe IDM
  - Laser Range finder (LIDAR): SICK LMS-200 (for navigation)
- Power supply
  - Uninterruptable Power Supply (UPS): APC Back-UPS ES 8 Outlet 550VA
  - Inverter: PI-800W Whistler CPI
  - Two Batteries : UB12260 Sealed Lead Acid (AGM)
  - Battery tenders: Battery Tender Junior 12 Volt 021-0123
  - LIDAR Power: 24V AC to DC converter
- Chassis
  - 3 levels: ½ inch thick 30inch diameter circles
  - 10 supports: Unbend aluminum
  - Two motor brackets: Machined aluminum
  - Two Caster brackets: Machined Plastic

- Locomotion
  - Two Motors: Brevel 420-652 12V with encoder attached to a gearbox
  - Motor Controllers: Pulse width modulated direct current motor controllers
  - External Board: ES308 Rabbit single board computer
  - Two Wheels: 6 inch diameter from storage
  - Two Casters: 3 inch diameter from storage
- Pitch control for Detector
  - Tilt apparatus: Machined aluminum
  - Linear Actuator: LINAK LA30.KL
- Control
  - Laptop computer: HP 8510W - Notebook

2.1.2 Construction

The actual construction followed the design described above, with a few minor changes. To save weight, the uppermost level was constructed out of plywood rather than aluminum. This was possible because the heaviest item on that level is the laptop computer. The total weight on the uppermost level is less than 10 lbs. Holes were drilled in the various levels to allow wires to be run between levels. A ½ inch offset between the driving wheels and the casters was inserted to aid the robot in tilting so that it can negotiate obstacles like going up ramps. Figure 2-2 is a diagram illustrating this.

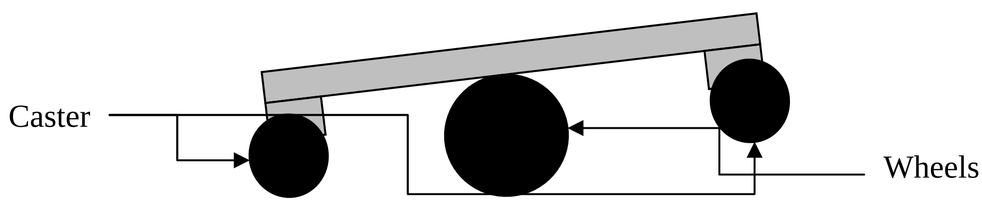

Figure 2-2. Caster and wheel diagram.

The initial design was to have the motors directly power the wheels, but this design was modified over concerns about shearing off the main motor shaft as had recently occurred in a circular design chassis. Instead it was decided to have the motors power the wheels through a chain, to allow for larger weight bearing shafts. During this construction phase, one gear shaft failed from fatigue, so the shafts in both of the Brevel motors used and a spare were replaced with new ones.

The two batteries (located on the lowest level) were held down with metal straps, and all excess cables were zip tied to the pillars or to tie down spots attached to the aluminum frame. A picture of the finished robot is shown below in Figure 2-3.

Figure 2-3. Completed robot.

2.2 POWER SYSTEM

The power system design was driven by the cooling requirements during cool down of the IDM. Due to the huge power draw (300+ W) of the IDM when cooling and to minimize system weight, it was decided to use wall power during cool down and battery power during system operation. Switching from wall to battery power required the design of an electrical circuit that contained an AC to DC inverter Uninterruptible Power Supply (UPS). The design was necessitated by the fact that if the IDM does not receive continuous power, it freezes and looses all communications with the laptop. It was therefore essential that the IDM receive continuous uninterrupted power.

2.2.1 Initial Design

In the initial design power came from the batteries through an inverter to a UPS. The UPS would then power all of the systems requiring AC power; the IDM, the Rabbit board and the LIDAR. In standard usage, a UPS is used to protect against surges from lightning and supply back-up continuous power to the computer while the power grid is down. In this application, the UPS' only purpose was to cover the conversion between wall power and battery power. Two switches were set up, one to switch the batteries from supplying the inverter to charging, and the other to control whether the UPS was drawing power from the inverter or the wall. Figure 2-4 is a diagram of this system.

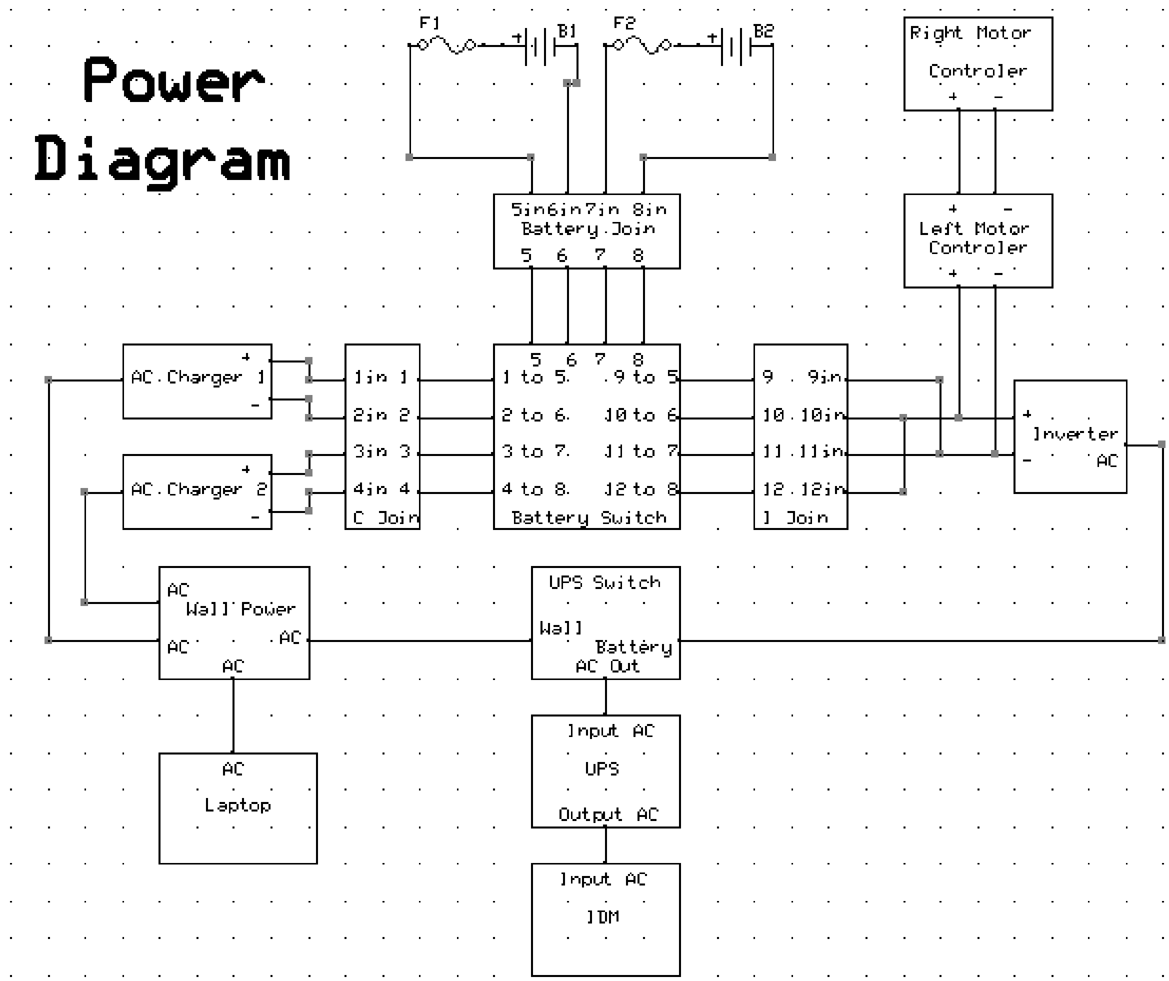

Figure 2-4. Original power circuit diagram.

Unfortunately this system did not function, for two reasons. First, the output of the inverter was not close enough to a sine wave to function as an input power shape for the UPS. Second, when the IDM first receives power, the current drawn by the system spikes briefly then settles down. This is no problem if the system is drawing wall power, however one time the system was drawing power from the batteries. This spike was sufficient to destroy the inverter. Given the existing problems with the power, it was decided that rather than try to repair the problem, the whole system would be redesigned.

2.2.2 Final Design

The final version of the power system does not require switches. In this version, the small battery on the UPS was replaced with the larger capacity batteries which had already been purchased for the system. The only downside of this arrangement was that if the UPS attempted to charge the batteries at the same time as the AC chargers, they would interfere with each other, because they would both misinterpret the other's power as the battery being fully charged. To prevent this, a diode was inserted in the circuit so that the IDM cannot charge them. A diagram of the final system is shown below in Figure 2-5 in which the diode is denoted with its electrical symbol and letter D.

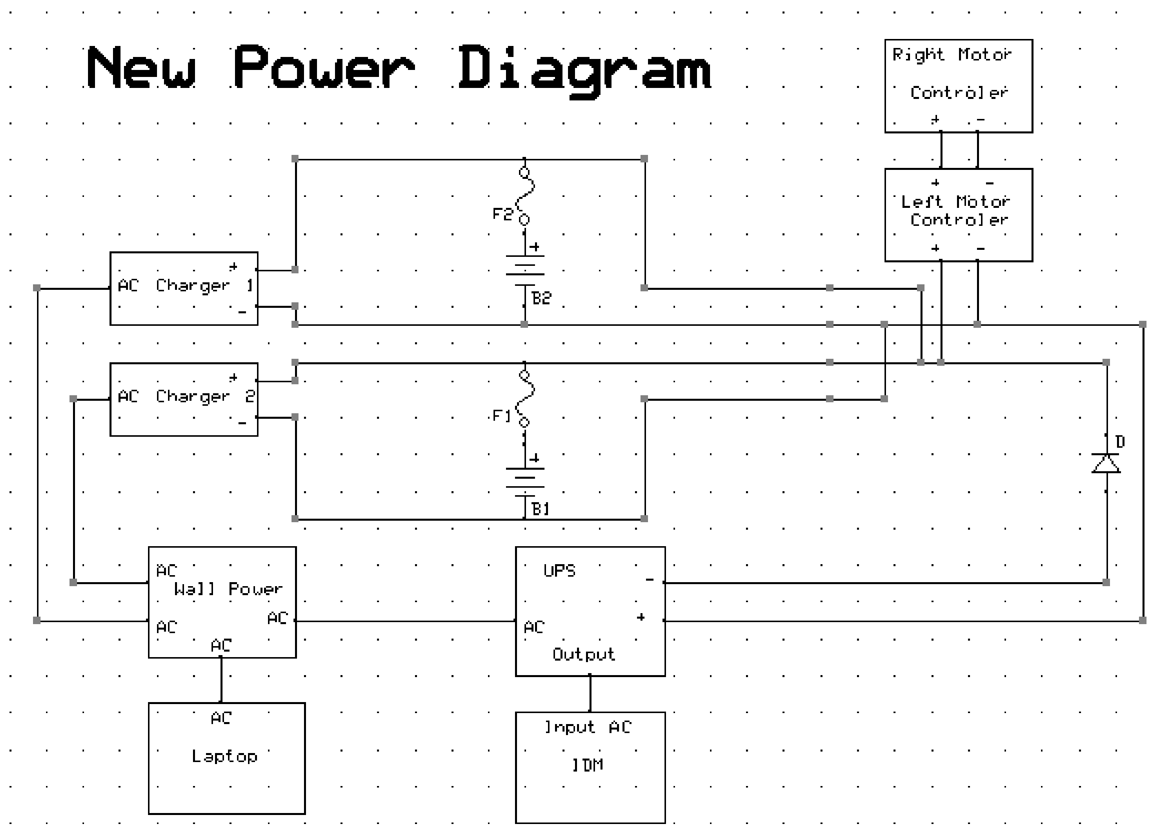

Figure 2-5. Final power circuit diagram.

When it was first implemented, the system shut down after an hour. This occurred because the newly purchased UPS had a timer-based self protection circuit which caused the system to shut down after an hour. However, when the UPS was replaced with an older model the system functioned perfectly and was now capable of supplying uninterrupted power for longer than the laptop can last on battery power alone. The advantage of this condition is that the laptop acts as a warning system for the main power supply before the system becomes too depleted and must be returned to a wall power.

2.3 MOTION CONTROL

The master control of the robot had to be performed in a modular fashion, because the LIDAR, the IDM and the motors all required some form of control input. A single program would be very overstretched if it tried to provide these various lower-level inputs at the same time as well analyzing the incoming nuclear and position data. Instead, a command structure is used, as illustrated in Figure 2-6.

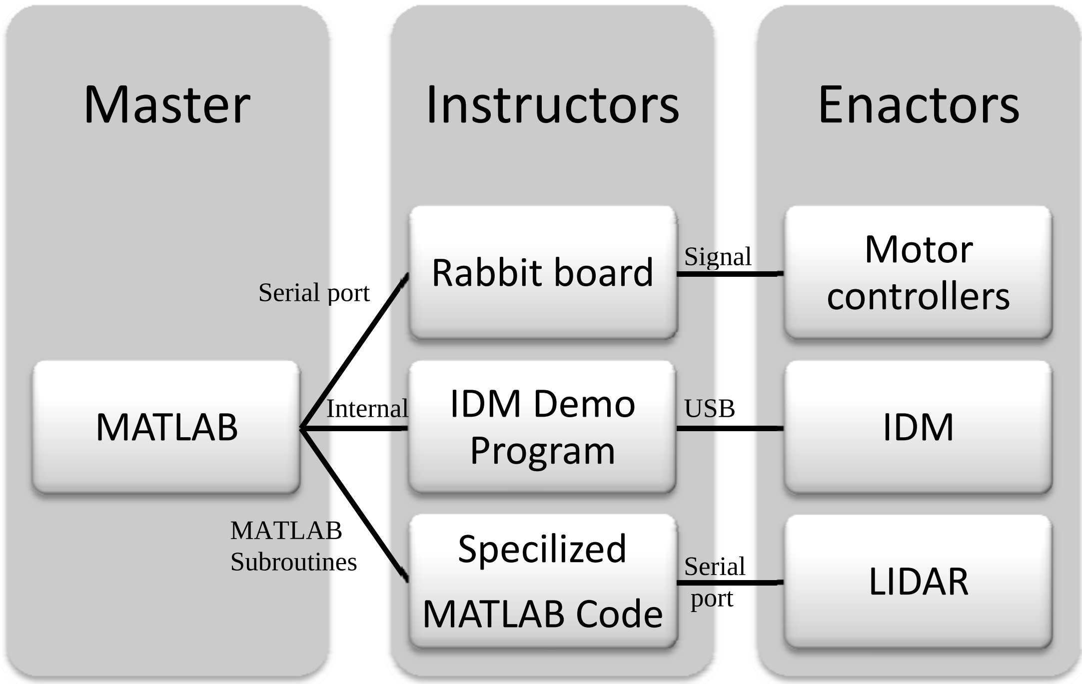

Figure 2-6 Dataflow diagram.

As shown in Figure 2.6, the final control of all these inputs was by a MATLAB program that had to be uniquely developed for this system. To accomplish this, all incoming data was formatted by gateway programs into a form MATLAB could handle, and in turn the MATLAB program sends out all instructions. When the master MATLAB program wishes to move the machine, it sends a signal down the serial port to the Rabbit board as shown in Figure 2.6. The Rabbit board then interacts with the Motor controller boxes and the encoders to actuate the instructions. When the master MATLAB program wishes to collect isotope information, a subroutine sends a signal down a network socket to the IDM Demo Program as shown in Figure 2.6. This program then interacts with the IDM and interprets the incident gamma radiation. Finally, if the master MATLAB program wishes to interact with the LIDAR, a different MATLAB subroutine and specialized code sends a signal down the serial port to the LIDAR as shown in Figure 2.6. The LIDAR interprets the results and then sends this information back to the Master MATLAB program. This approach was used to allow for more rapid prototyping and to keep the signal processing organized. The LIDAR and the IDM branches are discussed in more detail respectively in sections 2.4 and 2.5. The motor control section is discussed in the rest of section 2.3.

2.3.1 Basic concept of motion program

The motors used in this system are low speed, but are also high torque and provide very high precision positional information. An optical shaft angle encoder gives very precise information about the angular position of the motor shaft. This positional information is used in a closed-loop, feedback control system designed to precisely control the motor's position. As the wheel rotates, the shaft encoder translates a beam of light passing through a fan-blade shaped structure into electrical pulses representing shaft motion. The control system counts the pulses and therefore knows the position of each wheel. A complicated mathematical algorithm is used to quadruple the number of counts measured to improve accuracy, but the basic principle remains the same. The more sections on the encoder wheel, the more precise the control. Most wheels have 1024 counts per revolution or something similar. This motor only has 8 sections to the encoder wheel, 6 of which can be seen on the right of Figure 2-7.

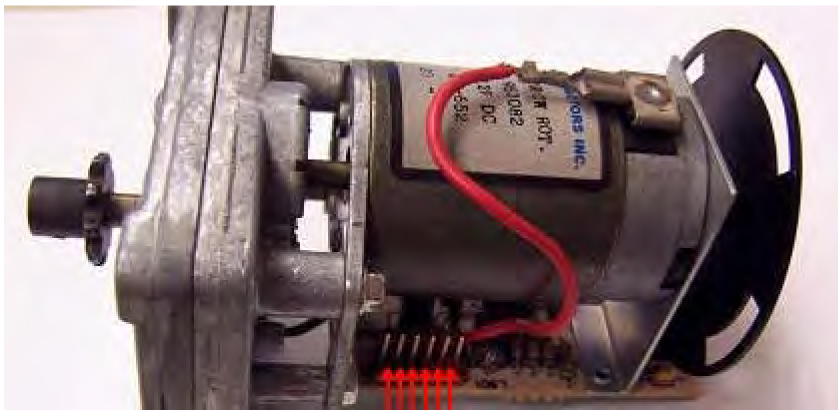

Figure 2-7.   Brevel 420-652 12V motor with encoder.

However, the encoder wheel measurement takes place before the attached gear box drops the speed. This means there are approximately 3200 counts to one revolution of the shaft out of the gearbox, giving extremely good precision. Because the motors are so slow, it was not felt necessary to feedback control the velocity of the wheels, only their relative positions.

To start the process, the MATLAB program sends a one to eight character instruction down a USB-Serial connection to the single board computer, nicknamed the 'ES308 Rabbit board' or 'Rabbit Board' as shown in Figure 2.6. It is a well established and well understood board within the systems department, with many useful functions that make it dramatically simpler to control a motor.

When the instruction reaches the Rabbit board it is interpreted and then implemented. For example 'f1x' means go forward one meter. The program written for the Rabbit then converts this signal into counts and an instruction to turn the right wheel forward and keep left wheel moving in the same direction. If the instruction had been 't90x' it would have meant turn 90 degrees left, which would have been converted into turn the right wheel forward and keep the left wheel moving in the opposite direction. The main program then waits until the right wheel has rotated the prescribed number of counts.

Keeping the robot moving in a straight line and keeping the turning radius to zero are the most important features of this control program. This lead to a novel approach to controlling the two motors. The right wheel becomes the 'master' and the left wheel is the 'slave'. All instructions are sent to the right wheel and decisions are made based on the progress of the right wheel. Roughly 100 times a second, a subroutine in MATLAB called an 'interrupt service routine' interrupts whatever is happening and checks the relative position of the left and right wheels. If the left wheel is behind the right, it increases the speed of the left motor and vice versa. A proportional integral negative feedback system is used to ensure that the speed of the left motor settles quickly. Once the right wheel has reached the set number of encoder counts, it stops moving the right wheel. It then waits until the encounter counts on the left wheel match the encoder counts on the right wheel, which is automatically accomplished by the interrupt within a few seconds. The program then resets the encoder count and informs MATLAB that the movement is complete by sending a string up the serial port.

2.3.2 Motion Calibration

Before the instructions could be given in meters and degrees, it was necessary to calculate how many counts corresponded to one meter or 1 degree. Though it is possible to calculate the number of counts in one revolution exactly and then use the radius of the wheel to calculate distance and degrees, this is not in fact the most accurate method. It is more accurate to directly compare the encoder counts output of the motor with the measured physical results of the robot's motion. For the distance measurement this was done by instructing the robot to move forward 32500 counts and then measuring the distance several times. The data used for calibration is shown below in Table 2-4

Table 2-4. Calibration for distance to counts conversion.

Counts	Distance (m)	counts per meter
32500	6.83	4758
32500	6.81	4772
32000	6.72	4765
32500	6.82	4765
32500	6.83	4758
	Average	4764

The final value used was 4760 counts per meter. For the angle measurement, the robot was instructed to turn 360 degrees on the spot at an estimated conversion. A straight line on the top of the robot was lined up with a straight line on the floor (in the Rickover Hall nucleonics laboratory) before this turn, then the difference between the two was compared after the rotation was complete. The conversion factor was adjusted until it was not possible to spot the difference. The final value used here was 28.11 counts per degree (programmed in as 253/9 counts per degree to allow the use of integers).

A final position feature was also added in the MATLAB program to study the motor's duty cycle. The duty cycle is the percentage of the incoming power which the motors receive. For the circular search it was initially thought that it would be useful to be able to turn at a certain rate, and be able to change that rate. A special MATLAB program with this ability was created which also outputs the current angle of the IDM at regular intervals, and had the unique ability to change what the robot is doing mid execution. This approach meant that the MATLAB program could send instructions to increase or decrease the robot's speed at any time. Using this MATLAB program, the relationship between the duty cycle (%) and the turn rate in degrees per second was found and the result is shown in Figure 2-8.

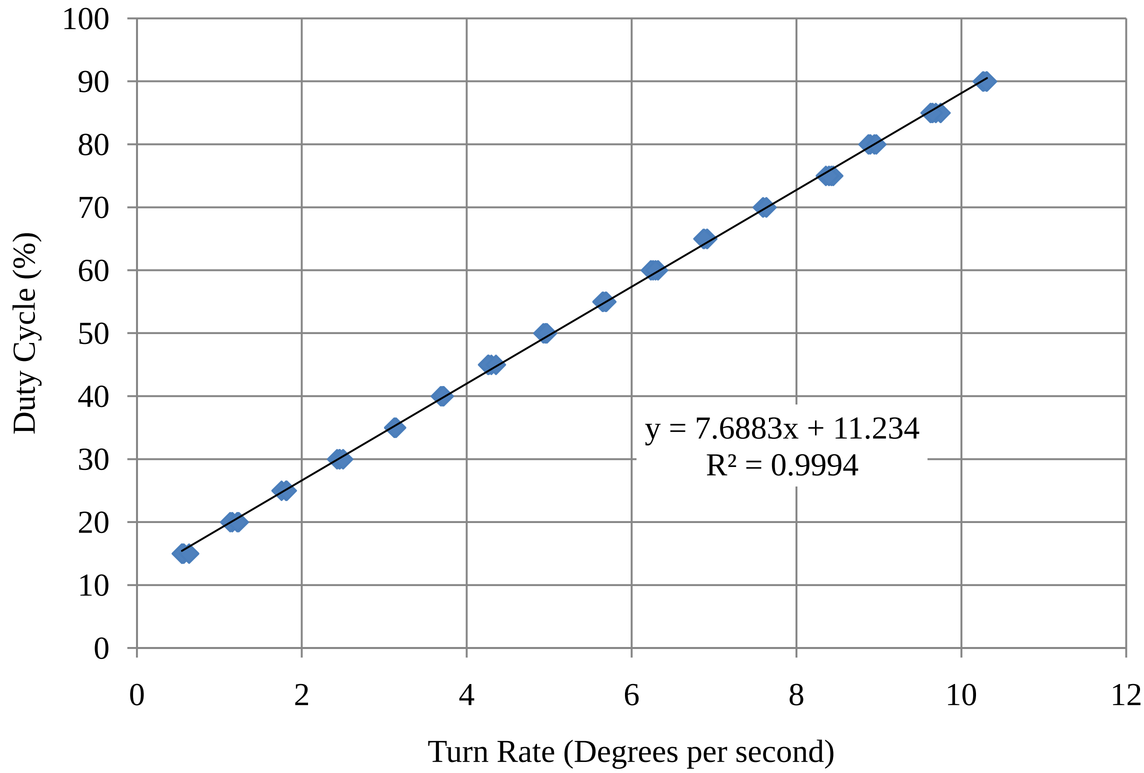

Figure 2-8. Calibration curve of duty cycle to turn rate.

As can be seen in Figure 2-8, a linear relationship exists between the turn rate in degrees per second and the duty cycle. Also shown in Figure 2-8 is s the result of a linear regression between turn rate and duty cycle.

2.3.3 Radio Control Motion

The second serial port in use on the rabbit board is attached to a 'Xbe Pro' radio transceiver, which can communicate with the joystick. After a short code is sent from the Laptop, the joystick can directly control the motors to allow the machine to be moved remotely. A diagram of the connections is shown in Figure 2-9.

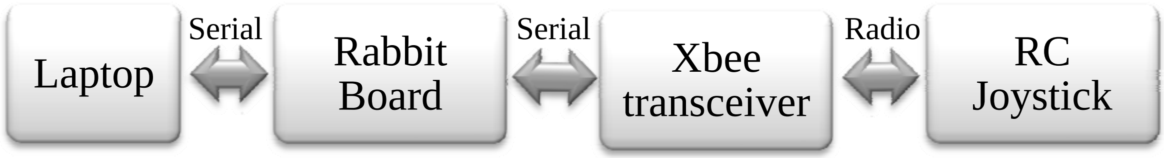

Figure 2-9. Diagram of communications used for radio control

2.4 LIDAR

The SICK LMS-200 LIDAR (shown in Figure 2-10) (SICK 2009) was used to more accurately calculate the change of bearing of the robot. With the use of the encoders on the wheels, dead reckoning based on encoder counts is effective over bearings of 90 degrees or more or distances of 0.5 meters or more. However, over shorter intervals gear and chain slop becomes a real issue. Every time the motors start, the tiny spaces between the gears and a loose length on the chain lead to inaccuracies between the encoder wheel attached to the initial drive shaft and the final wheel touching the ground. This means that while the system can turn 360 degrees and maintain a half a degree of accuracy (depending on the floor), if you program it to turn 5 degrees 72 times, it will rotate somewhere closer to 400 degrees (i.e. a 40° error) once the system comes to a stop because less than half of a degree of error was introduced every time the motors started or stopped. To counter this problem, the LIDAR range finder calculates the estimated bearing, and this information was used to locate the position of the nuclear source, rather than the initial dead reckoning.

Figure 2-10 SICK LMS-200 LIDAR

2.4.1 Summary of SICK LMS-200

The SICK LMS-200 is a 75 hertz laser scanner capable of delivering range data over a 180 degree field of view in half degree increments (SICK AG 2009). The LIDAR is mounted on the top level of the robot, so that nothing could obstruct its field of view. Fortuitously, the top level is made of wood and the SICK LMS-200 has screw holes in the bottom, allowing the system to be easily secured. The LIDAR is supplied by an AC to DC 24V power supply on the top level. The LIDAR's original connector end was removed and replaced with a Serial connection in order that it could interface with the MATLAB program.

The data from the SICK comes into the MATLAB program as a series of lengths, corresponding to the range measurement taken at half a degree intervals. These can be constructed into an image of the surrounding area (as shown in Figure 2-11) or can be left simply as a set of lengths (as shown in Figure 2-12)

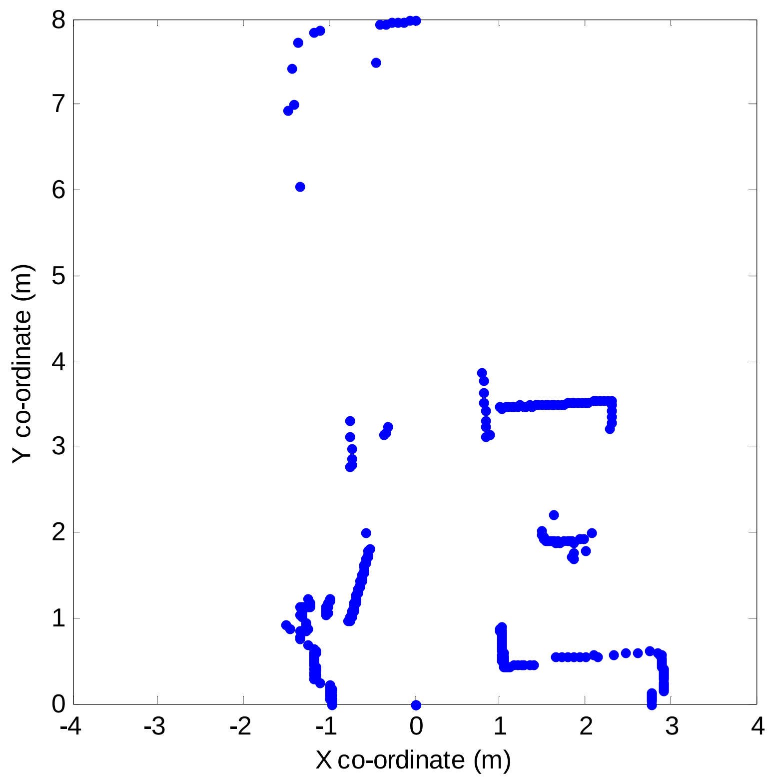

Figure 2-11. Output of the LIDAR in XY format.

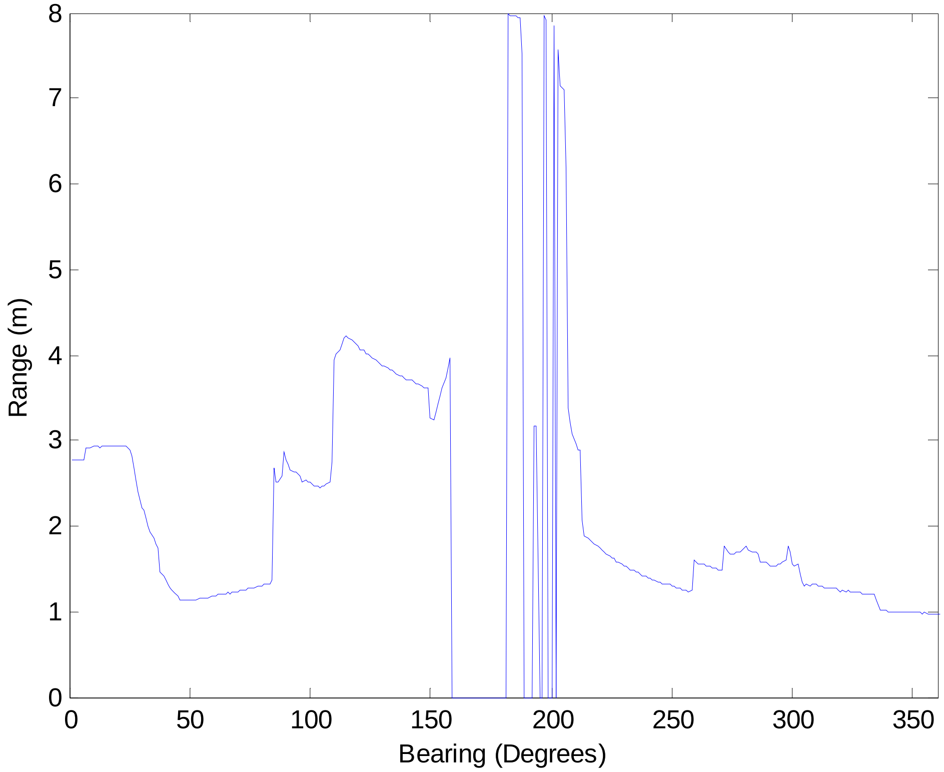

Figure 2-12. Output of the LIDAR in range-bearing format.

2.4.2 Bearing Calculation

To calculate the change in bearing, two readings are compared in the form of a series of distance measurements. One reading is shifted across the other and the mean square of the differences between the two readings is calculated for half a degree of shift. For the initial reading, zero degrees is compared with zero degrees and so on. On the next iteration zero degrees is compared with the half a degree reading on the other. This process is repeated until a comparison for four times the estimated angle of the vehicle is completed. The mean square difference for each shift of one set of distance readings relative to another is then recorded. The shift with the minimum mean square of the differences is assumed to be the most accurate measurement of the change in bearing of the robot.

Due to time constraints, the decision was made not to include direction correction. Instead, a careful record is stored by the system of the LIDAR results at each place where the readings were taken. After all of the readings are complete, the master program goes back and calculated the bearing at each point. This is done by comparing each reading with the three readings taken before to arrive at an average estimated change in bearing. Each bearing builds upon the previous reading to give an overall estimate of each position. The bearings calculated by the LIDAR are used when plotting the results of the IDM measurements. The accuracy of the final source bearing calculation is completely dependent on the accuracy of the calculation of the bearing of the robot. If the robot calculation is off, so is the final result.

2.4.3 Characterization of Accuracy of Bearing Calculation.

To calculate the reliability of the bearing calculation, observed results were compared with calculated positions. The machine was lined up with the edge of a floor tile in the Rickover Hall Nucleonics Laboratory, then programmed to turn in 1 or 2 degree increments and calculate the new position. The calculation of the actual position was only conducted after the move was complete; however the nominal bearing calculated by dead reckoning was displayed on the screen at each increment. This was used to find out which calculated bearing corresponded to the instant when the machine was facing 90 degrees from where it started. The results are shown in Table 2-5.

Table 2-5. Characterization of accuracy of bearing calculation.

Actual Bearing (degrees)	Dead Reckoning Bearing (degrees)	Dead Reckoning Bearing Error	LIDAR Calculated bearing (degrees)	LIDAR Bearing error
85	61	28.2%	85.62	0.73%
90	76	15.6%	94 .00	4.44%
90	78	13.3%	95.14	5.71%
90	65	27.8%	92.96	3.29%
90	62	31.7%	84.85	5.72%
90	66	26.7%	91.00	1.11%
110	83	24.6%	111.96	1.78%
110	84	23.6%	115.00	4.55%
110	79	28.6%	115.50	5.00%
110	79	28.2%	114.60	4.18%
180	150	16.7%	189.00	5.00%
270	202	25.2%	263.00	2.59%
	Average	24.18%		3.68%

From this it can be seen that the calculated bearing is never more than 6% off of the observed bearing. Though this is not perfect, it was sufficient for this project. When plotting the results of a reading, a marker for both the intended position of the source and a 6% error bar to account for the LIDAR error was plotted, as the location of the source can only be as accurate as the LIDAR estimated position of the robot. This error is further discussed in sections 4.1.3 and 4.1.4 where the bearing characterization of the system and its ability to determine source bearing is given.

2.5 ORTEC IDM DATA INTERFACE WITH MATLAB PROGRAM

This section describes how the ORTEC propriety software was interfaced with the MATLAB program to determine the identity of the radioisotope as well as its location within a room. The IDM comes with its own propriety software, which displays any radioisotopes it finds. This software was not directly compatible with MATLAB and thus a program had to be written to convert its information into a form the MATLAB program could use.

To determine the presence of a radioisotope, the IDM software calculates a statistical parameter called the Q value (see eqn 1.1) for all 71 isotopes and material in its gamma spectroscopy library. The IDM Demo Program software (written by Mr. Randy Meyers of ORTEC) calculates the Q values every ¼ second for a prescribed time called the window length. A window length of 5 seconds would then consist of 20 readings. The identified isotope and its Q value is then passed from the IDM Demo Program to MATLAB using a simple C program which does little but translate the data into a form which MATLAB can use easily.

Once within MATLAB, each isotope identified by the software is compared with a list of isotopes identified previously in the overall reading. Each isotope is then identified by the MATLAB script with a number, for example the first isotope identified is given the number ‘1’. Using that isotope number, the Q value and count number of that isotope is recorded for that bearing and then the process is repeated. Thus a Q value is stored for all isotopes every ¼ second, which implies that these repeated readings are not independent. Using this method, the Q value at any given bearing for any given count number (i.e. repeat) is stored in a retrievable fashion, distinct from all other readings.

Figure 2-13 gives a flow chart of how the circular search program, called “SourceFinder” works, bringing together all of the lines of communication in sequence as needed.

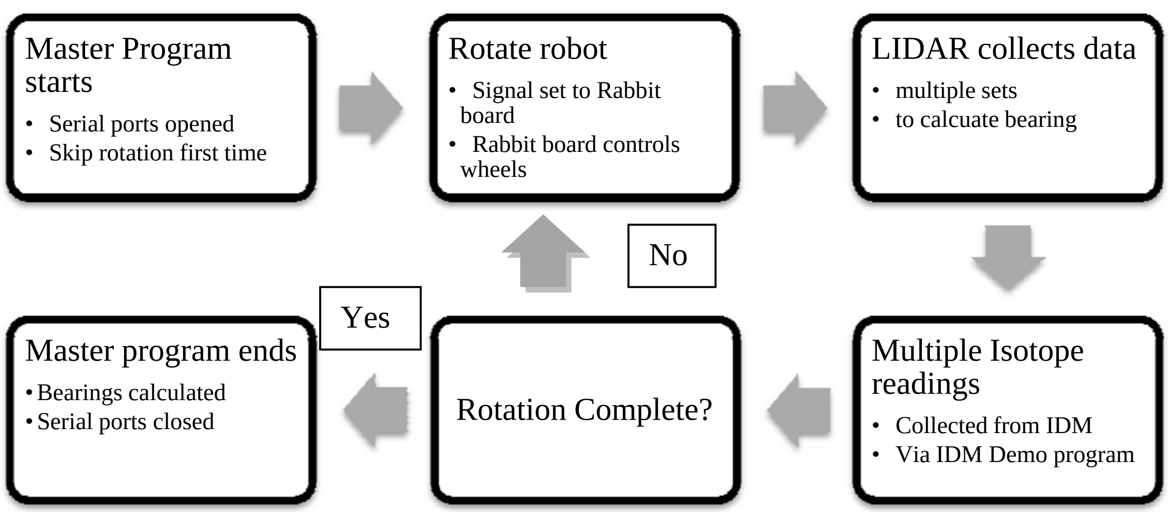

Figure 2-13. Flow diagram of a circular search

SECTION 3

RESULTS, CONCLUSIONS AND RECOMMENDATIONS

3.1 RESULTS

The ability to detect the bearing of a desired source is not as trivial as it first sounds. The estimates of the bearing of the robot have to be carefully calculated and the data coming from the IDM is extremely noisy. Using repeated readings and filtering processes, it is possible to estimate the bearing of the source, but there were many stages involved in creating that algorithm.

3.1.1 Time Window Characterization

When calculating the probability of sources being present, the IDM uses a specific amount of data. Data is collected in 0.25 second increments. Any number of these increments may be used in a calculation. This means that sources can be identified based on as little as 0.25 seconds of data, or as many as 8 seconds of data or more. The total data collected each time which is used to calculate the Q value is called the time window. If the time window is lengthened, the results are more accurate. However, this process can be very time consuming. It is important to balance input time and accuracy.

For the following data, a 0.25µ Ci Cs-137 source was directly in front of the robot, with the sensor pointing in a direction of relatively low background noise. The time window used to calculate the Q value and the distance between the HPGe sensor and the source was then varied, and 100 independent readings were taken in each setup. A diagram of the setup is shown below in Figure 3-1.

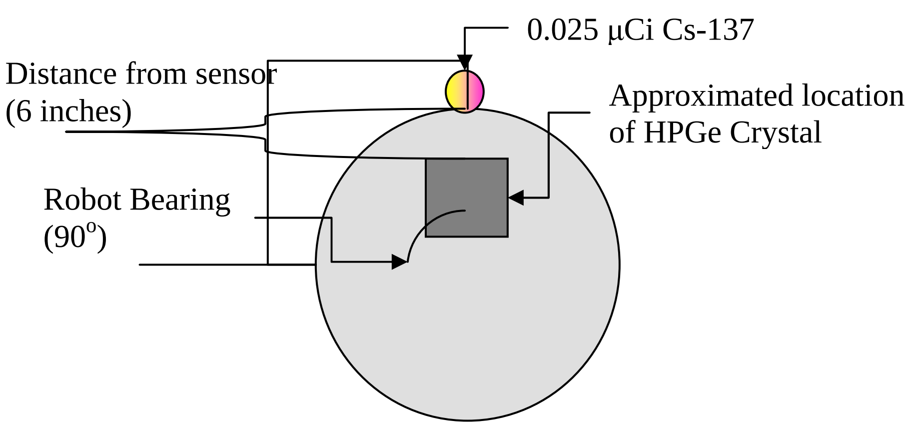

Figure 3-1 Set up for time window and distance characterization

It should be noted that a ‘sliding window’ is normally used for collecting repeats of the same data. This means that if the IDM is calculating based on an 8 second window, then every quarter of a second, the oldest quarter of a second’s data is thrown out and the newest is thrown in and a new isotope identification is performed. This means that the new repeated reading is not completely independent of the previous readings. The following characterization data consists of 100 independent readings, so each reading doesn't share a single set of input gamma ray data with any of the previous ones. This is more time consuming than using readings that are not independent, but it seemed appropriate for characterization purposes. The sliding window is more time efficient, but the independent readings will give a better picture of the sensor's capabilities.

The data was collected over a variety of days, and it quickly emerged that the Q value results for the same set up varied from day to day. This can be due to the level of background radiation that day, how long the HPGe crystal has been cooling, location of the sensor when the data is collected, alignment of detector and source and other effects. Data from identical setups from different days cannot be plotted on the same graphs, with one exception, because the results are too different. However, it is still possible to deduce some broad generalizations and trends, and even one correlation which held true any day.

Throughout the process it is important to remember the two boundary values of Q. A Q value below two is taken to mean there is no source present, so a result lower than that is never collected. A Q value above five is supposed to correspond to a strong certainty, and so its location will be marked on each graph in this section with a purple line.

The first of these conclusions is best shown with something called a box plot. Each column represents 100 pieces of data. The red line shows where the median is, and the space between the red line and the lower blue line represents a quarter of the data, and the blue line is called the 1st quartile. The next quarter of the data is usually found between the blue line and the end of the black "whisker", unless the data is an outlier. The space between the median and upper blue line represents another quarter (called the 3rd quartile) of the data and so on. Half of the data is contained within the box, and the distance between the 1st quartile and the 3rd quartile is called the inter-quartile range. An outlier is defined as any data point which lays more than 1.5 times the inter-quartile range away from either the 1st or the 3rd quartile. Outliers are instead represented as red crosses beyond the range of the whisker. It should be noted that the inter-quartile range is roughly proportional to standard deviation, in that both represent measures of the spread of the data. Notice how the size of the boxes in Figure 3-2 does not decrease as the time window is increased. Increasing the time window did increase the mean Q value, but it did not reduce the inter-quartile range of the data, and thus its spread. The conclusion from this is that longer read times do not lead to more precise data from the IDM.

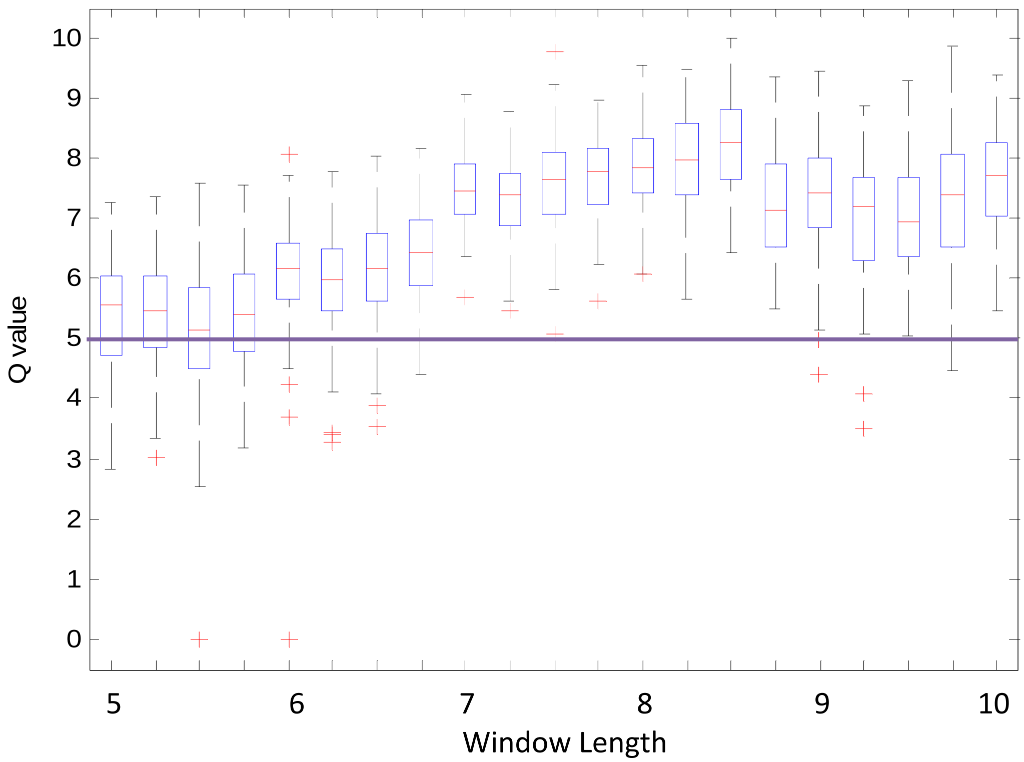

Figure 3-2 Spread of Q value versus window length, Q>5.

As shown in Figure 3-2, above a Q value of 5, there is not a clear linear correlation between time window and Q value. This held true in other sets of data which are not shown. However, when the Q value is below 5, there is a very strong correlation between time window and Q value. Figure 3-3 shows a plot of the Q value versus the time window with Q values less than five. The exact relationship of the correlation in Figure 3-3 linking window length to mean Q value only holds true for the specific set of data plotted. However, while the Q value is less than 5 there is a strong linear relationship between the two, but the exact relationship varies from day to day. This variation is caused by daily changes in background, how long the HPGe crystal has been cooling down and other parameters.

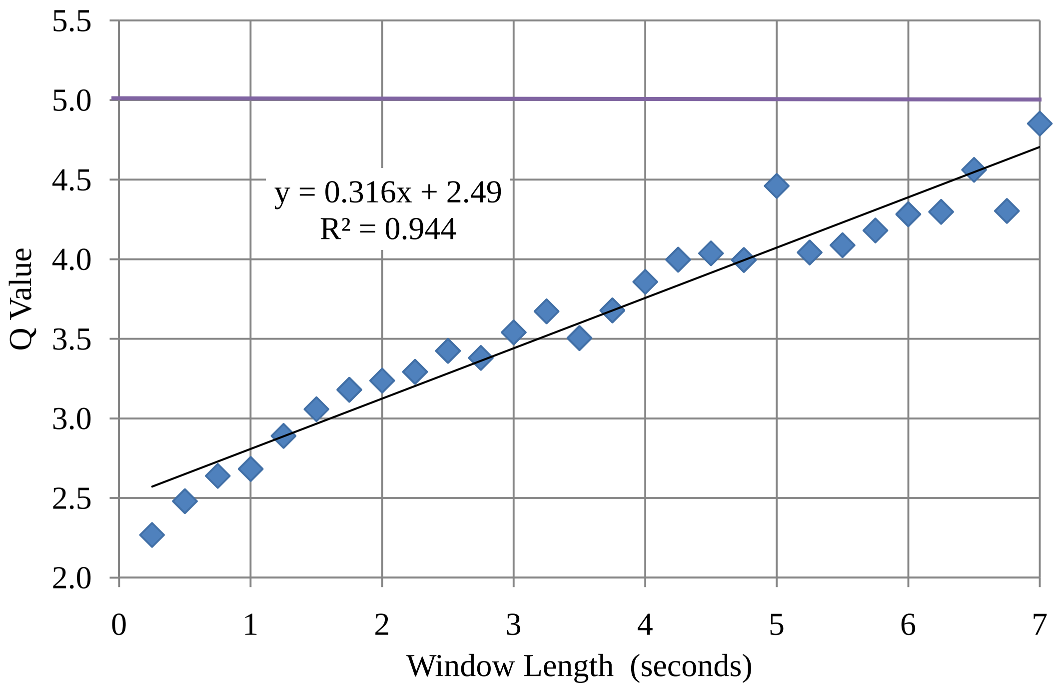

Figure 3-3. Relationship of Q value versus window length when Q<5.

The IDM does not always sense the presence of the source, or if it does, the Q value is less than 2.15 so it is considered negligible, even if it present. This is known as a false negative, because it calculates the source is not there when it is. The number of times it does not do this per 100 readings is known as the detection rate. The lower the detection rate, the greater the number of false negatives in that data set. There is a strong exponential relationship between the detection rate and the time window, shown in Figure 3-4. The coefficients in the equation shown in this figure were found to vary slightly from day to day, but while the detection rate is not 100%, the detection rate of the Cs-137 in front of the IDM was always found to be exponentially related to the window length.

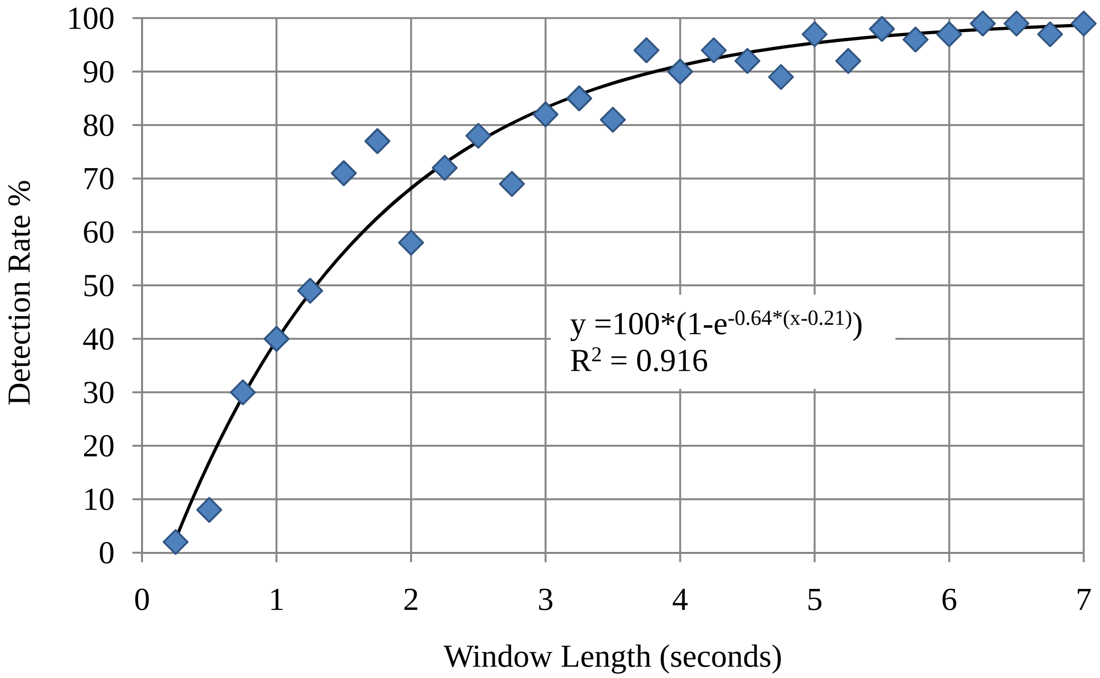

Figure 3-4. Correlation between window length and detection rate.

ORTEC uses a Q value of 5 or greater to indicate isotope identification. Based on this recommendation, a time window of 7 seconds was selected for further study. Over many days, many sets of data in similar set ups as shown in Figure 3-1 were collected, and average Q value was found to fluctuate even with same window length. However, 7 seconds was the shortest time window which consistently detected the source at a Q value of five or greater and a 100% detection rate.

It should be noted that this 7 second window was based upon data collected from a 0.25 μCi Cs-137 source. Though there were plans to study this response with other sources of different strengths in order to create a more generalized time window criterion, the project ran out of time to collect the data necessary to justify a universal statement on the optimum time window setting. In the following sections, with one exception, a 7 second time window was used when collecting data. When searching for stronger sources, it should be possible to use a smaller time window and still locate them with an associated Q value of over 5.

3.1.2 Distance Characterization

Once the effect of the time window was studied and understood for this set up, the effects of distance were examined. As before, 100 independent readings were used with a seven second time window. By independent reading, it means the Q value was studied every 28th reading as the IDM software collects data every ¼ second. The results of this study are shown in Figure 3-5.

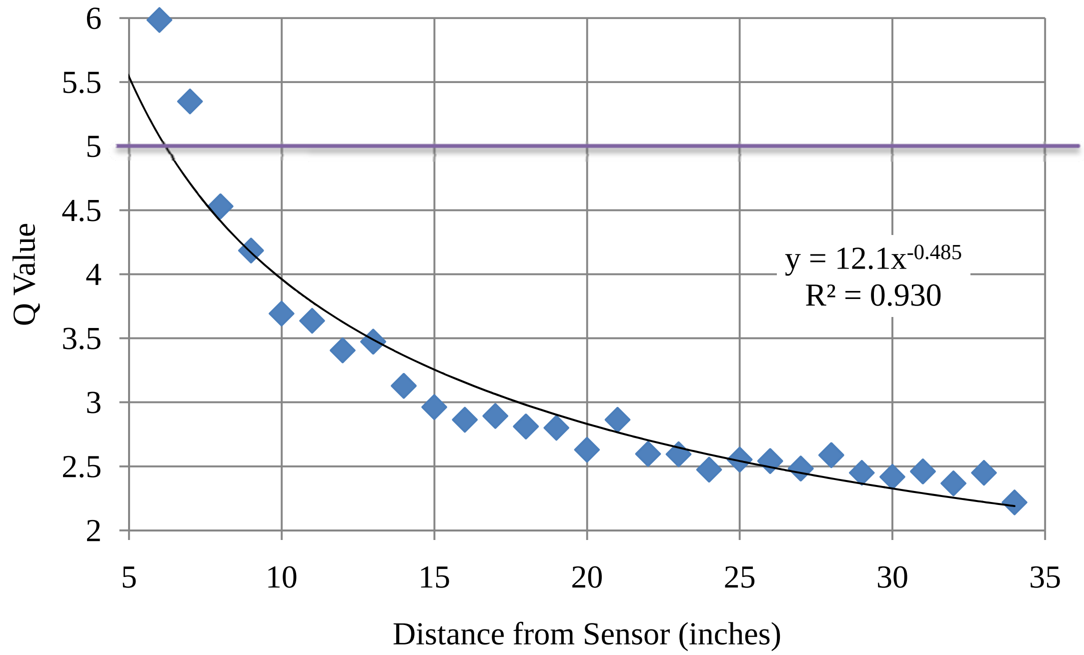

Figure 3-5. Inverse square root relationship between Q value and distance.

As Figure 3-5 shows, there is a strong relationship between distance from sensor and Q value which is approximately equal to $1/\sqrt{distance}$. Though a relative increase in the Q value is achieved by decreasing the distance between the source and the detector, the exact relationship would require an understanding of the source/detector geometry. The detection rate relative to distance was also found, as shown in Figure 3-6.

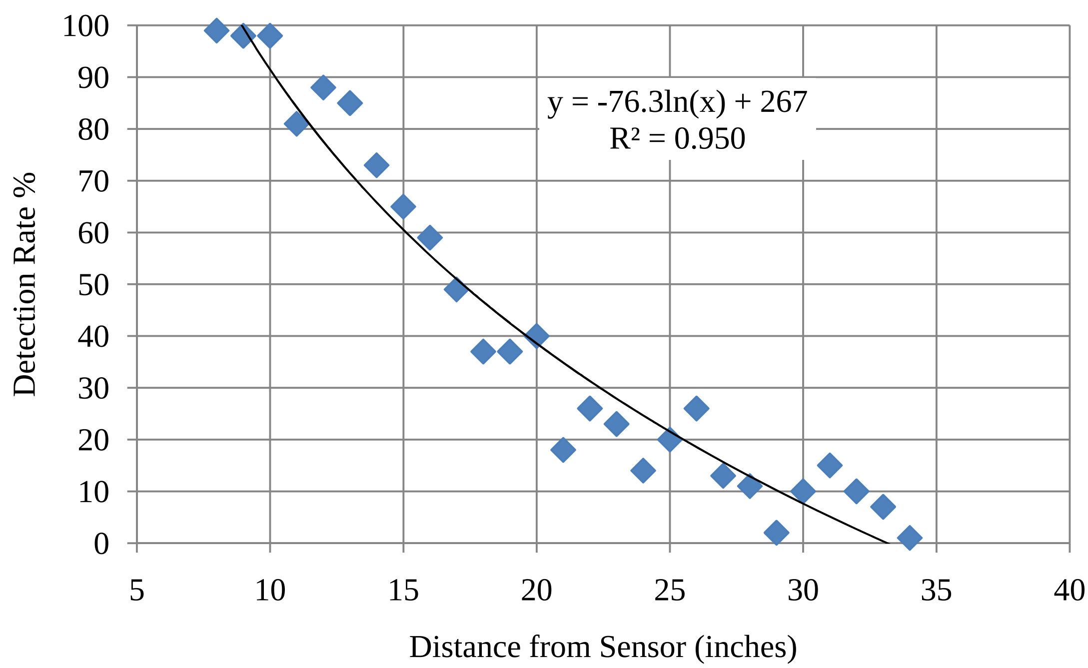

Figure 3-6. Relationship between detection rate and distance.

The most clear conclusion that can be drawn from Figure 3-6 is that the detection rate falls as the distance from the sensor increases, which was expected.

There is one last correlation found in this study. Using the time window and distance characterization data from several experiments (some not shown), an empirical correlation between the detection rate and Q was found. This correlation is shown in Figure 3-7

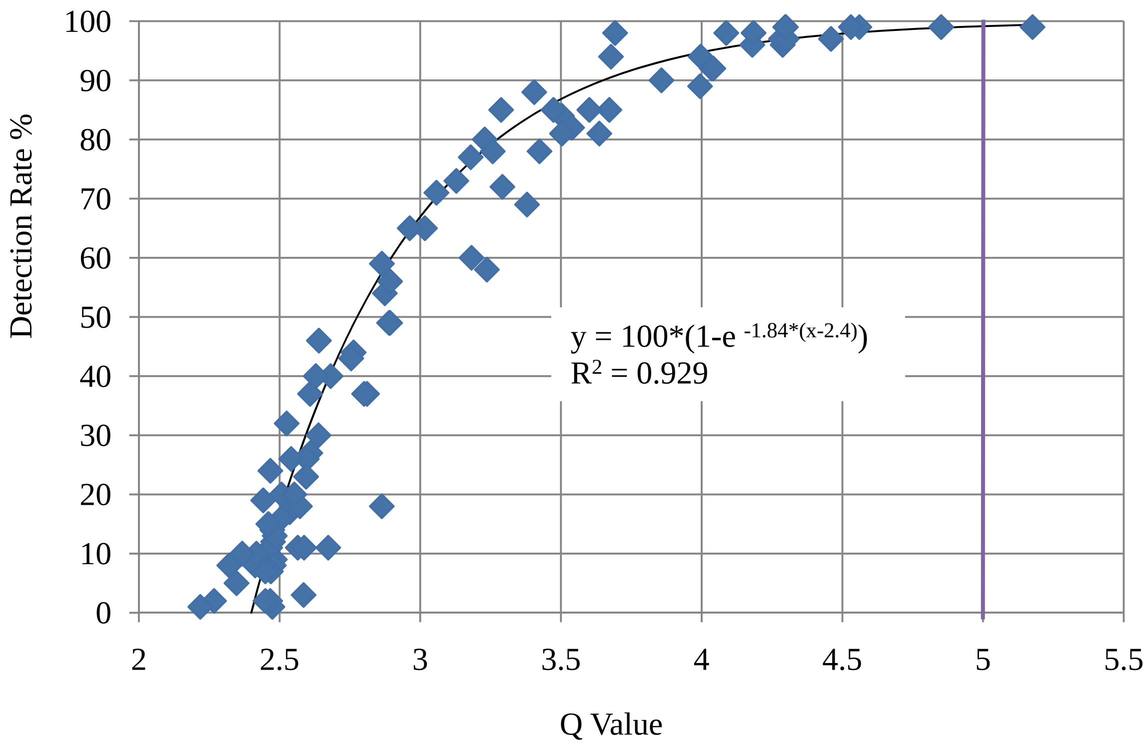

Figure 3-7. Relationship between detection rate and Q value.

The equation for this correlation, which held true through multiple different days, is:

$$DR = 100\left(1 - e^{-1.844(Q-2.4)}\right) \tag{3.1}$$

where

DR = Detection Rate

Q = Quality factor or Q value

Figure 3-7 justifies the ORTEC's choice of 5 as a Q value for isotope identification. Above a Q value of 5, a detection rate below100% is extremely rare. This is possibly the most exciting result of the study. The Q value used is in reality the mean Q value of the successful readings, so to see that the detection rate has such a strong correlation with such a random variable was not expected.

3.1.3 Characterization of effect of bearing to source

To determine the effect of the bearing of the detector relative to the source, a further set of dedicated readings were taken, shown in Figure 3-8. To collect the data a 32 second window was used, to increase the Q value, and the robot theoretically turned 1^{o} per reading. In reality, the turns were slightly bigger than this, but the LIDAR was used to calculate the bearing at each source, and that number was used in plotting the bearing of any given reading below. Only 8 readings were taken at each bearing, and these readings were not independent. The bold green represents the intended location of the source, in this case 110^{o}. The two thinner green lines represent the error bar for the LIDAR. The actual location of the source was somewhere between the two thinner lines.

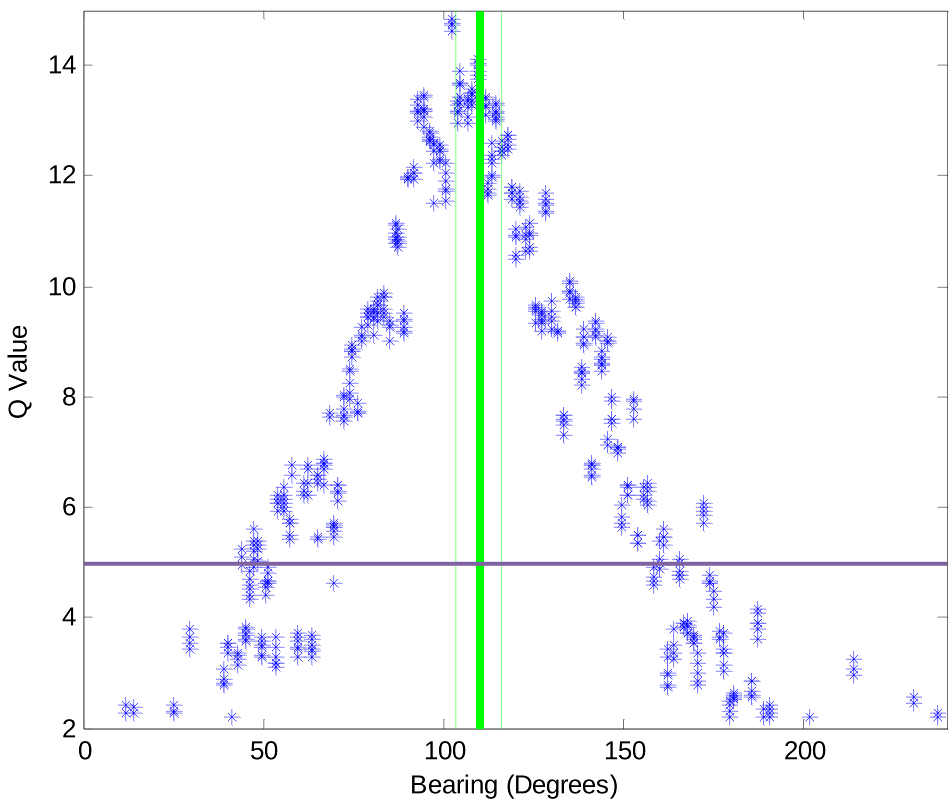

Figure 3-8. Plot of mean Q value versus window length, no fit.

As Figure 3-8 shows, the Q value increases in a triangular shape as the detection moves into the correct bearing. To locate the source, it is necessary to curve fit accurately to a triangle shaped peak in the readings.

Initially a fourth-order polynomial curve fit was used to fit the correlation between the bearing of the robot relative to the source and the Q value. This can be seen in Figure 3-9, which shows that data from Figure 3-8 with a 4^{th}-order polynomial curve fit shown in red. Notice how the peak (which is supposed to correspond to the location of the source) is slightly to the right of where the human eye estimates the maximum ought to be, even though it lies on the intended location of the source. This is because the 4th order fit is being pulled to one side by slightly skewed data.

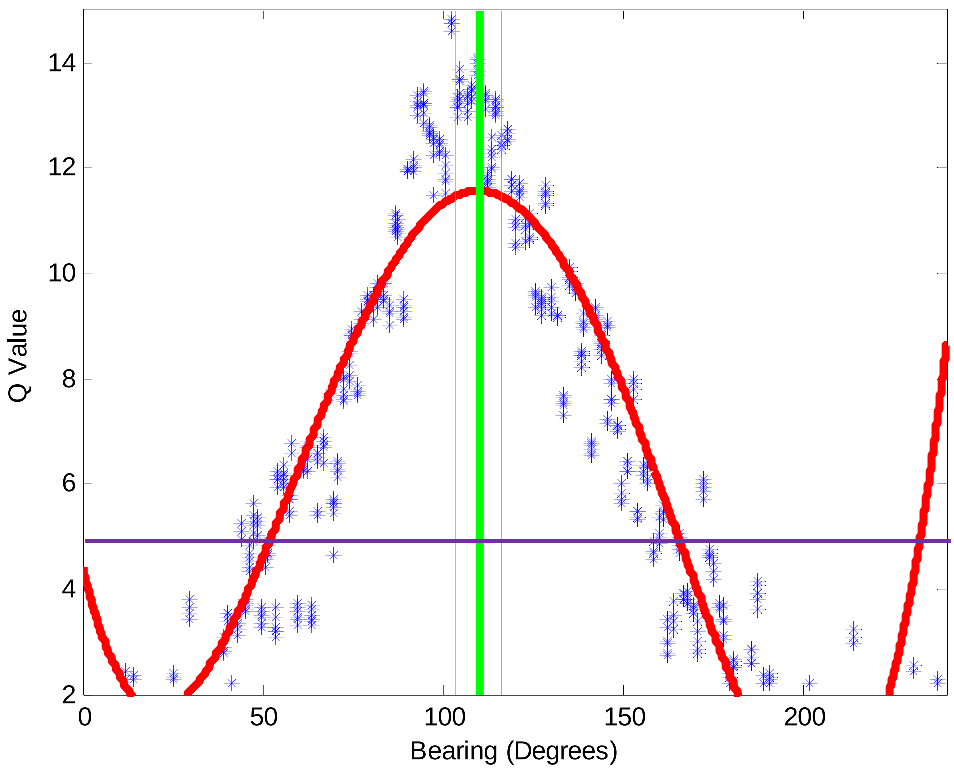

Figure 3-9. Plot of mean Q value versus window length, 4th order polynomial.

Next the Q value versus bearing was fit using two separate linear lines. To do this the maximum Q value was used as an initial estimate of the location of the source. The data from fifteen degrees either side of it was discounted and the remaining data split into two halves. A linear line was then fitted to each half, and the place where the two lines intercepted was calculated as the location of the source. The result is shown in Figure 3-10, which shows the same data as Figures 3-8 and 3-9, but this time with a double line fit on top in red.

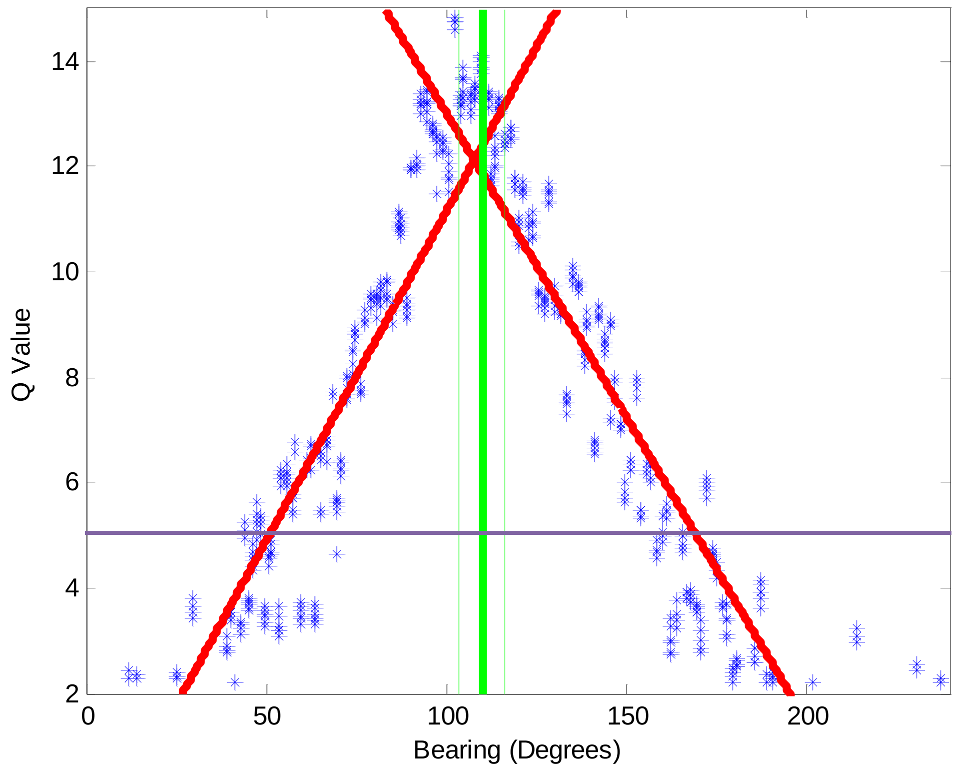

Figure 3-10. Plot of mean Q value versus window length, double line

As shown in Figure 3-10, this curve fit predicts very close to the actual source location of 110°. It should be noted that an initial filter was used on the data, removing all Q values below 5 to reduce of noise that is inherent in the initial detection of a source. When the same fit was used on other 32 second data sets, the results were also extremely satisfactory. The decision was therefore reached to use the double line intercept fit for calculation of the location of the source.

3.1.4 Optimizing program settings of bearing interval and repeats

When this stage was first reached, it was assumed that after a certain point, you would reach a point of diminishing returns, where increasing the time would not dramatically increase the accuracy, and that could be declared the optimal point. This was not the case. Multiple sets of data were taken, using a seven second time window of data for 15 seconds, which gave give 60 data points at each degree through a 180 sweep. A program was then written to calculate the location of the midpoint, according to the curve fitting algorithm developed in section 3.1.3. An experiment was then performed using the set-up in Figure 3-1 and the location of the source calculated multiple times, varying the number of the 60 data points readings used and the interval in degrees (°) between bearing readings. The percent error of the calculated source bearing relative to the true bearing of the sources versus the time taken to collect the data, and the result is shown in Figure 3-11.

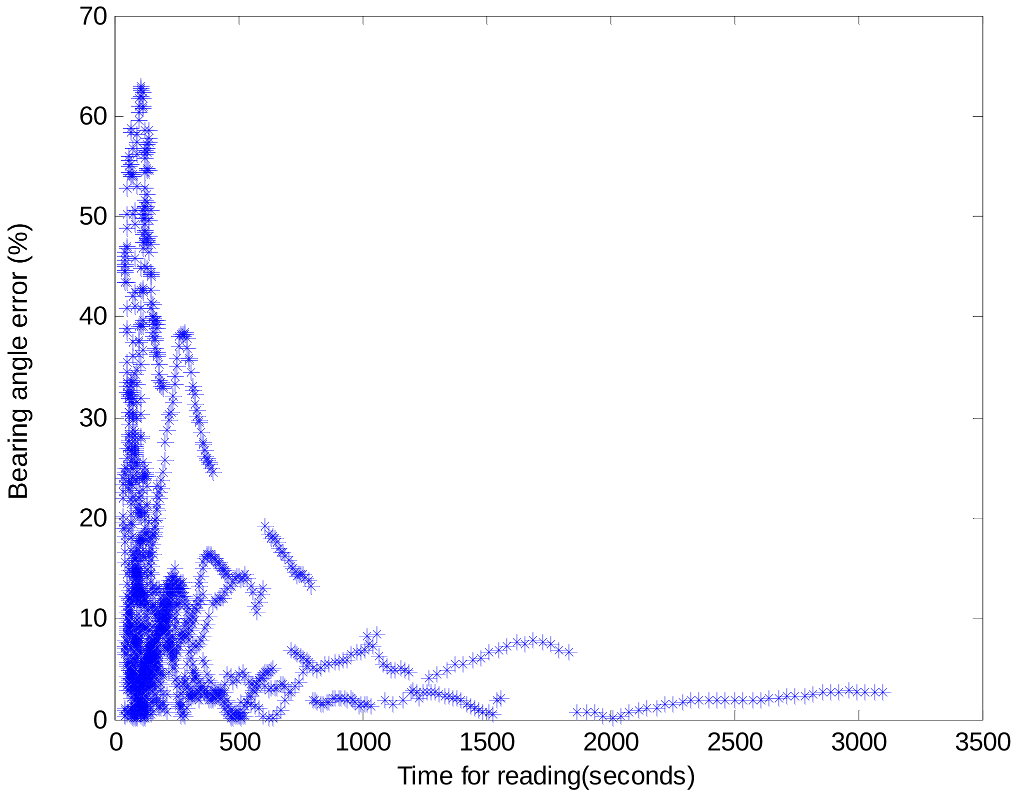

Figure 3-11. Bearing Error verses time for reading.

Figure 3-11 shows that maximum bearing error was greater when the reading time was below 1000 seconds compared to when it was greater than 2000 seconds. Thus the maximum error possible decreases as the time for reading increases, but it appears that even for very long reading times a certain bearing error will always exist. On most sensors, a plot like this would show a minimum time length required to get a certain error. Instead, an accurate result is possible with almost no time, and rather than picking out the setting required to achieve the minimal error, all that is possible is to calculate the maximum error for the time taken.

A plot of the calculated source location versus the number of readings used and the degree intervals used, shown in Figure 3-12, gives some explanation for this. Normally such a plot would be expected to steadily converge on the correct answer as more readings were added and the interval between readings was reduced. This did not occur.

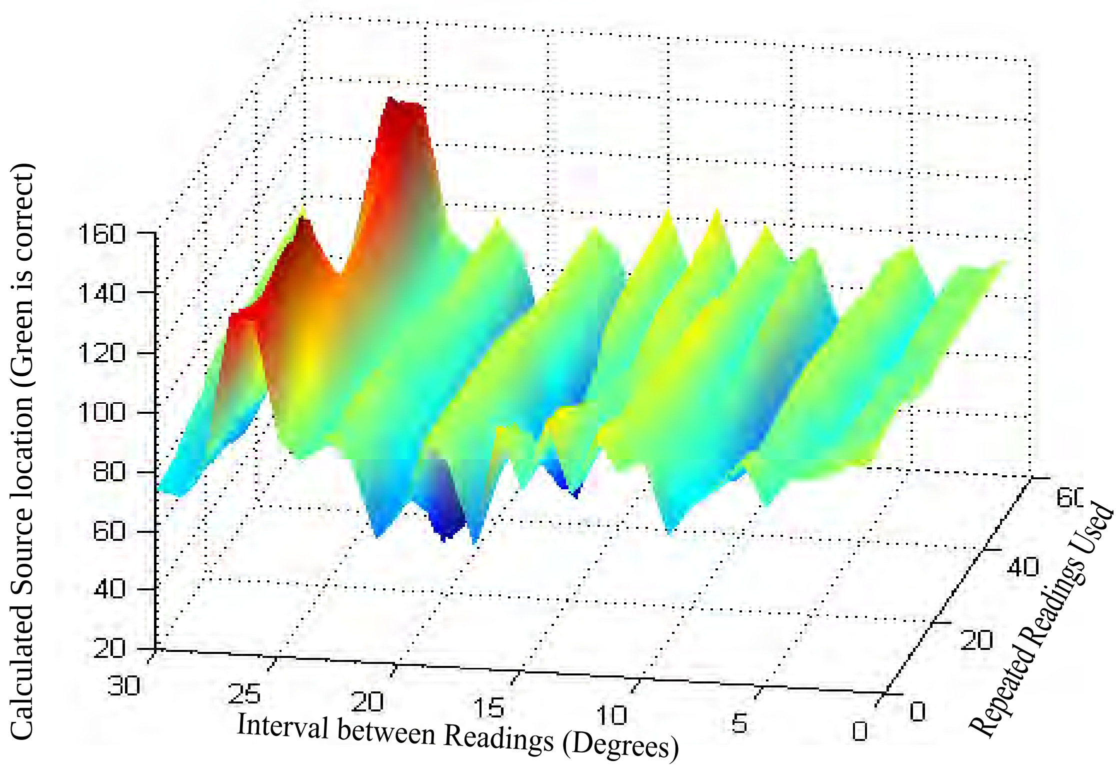

Figure 3-12. Plot of source location versus degree interval and repeats, surface.

In Figure 3-12, the results change color according to the value of vertical or Z axis. Ninety degrees (the correct answer) is shown in a pale green color. A red color means that the calculated result is too high, a blue color means that it is too low. Clearly there are large amounts of variation in the data. One conclusion which can be drawn from this graph is that the interval between readings cannot be higher than 15° or the bearing location will likely have a large error. However, it should be noted that decreasing the bearing interval below 15 doesn't seem to improve the results. Instead, at around a 15 degree interval a point is reached where the natural variation of the system has more effect than the angular interval between readings.

Another conclusion drawn from this study is more easily shown with a ribbon plot. A ribbon plot is like a series of line plots, each in a different color, where if follows one variable as it changes through two dimensions. The first figure, Figure 3-13, clearly shows a set of independent lines when the bearing interval is held constant and the number of repeated readings is varied. The second figure, Figure 3-14, shows a series of lines where the number of repeated readings is held constant as the interval between readings is changed. Since these lines move almost in parallel, it suggests that the intervals between the bearings has a much stronger effect on the correct bearing reading than the number of readings (i.e. time) used to calculate bearing position. From this study, it was concluded that it doesn't matter how many repeat readings along the same bearing are used, only that there is at most a 15 degree interval between them.

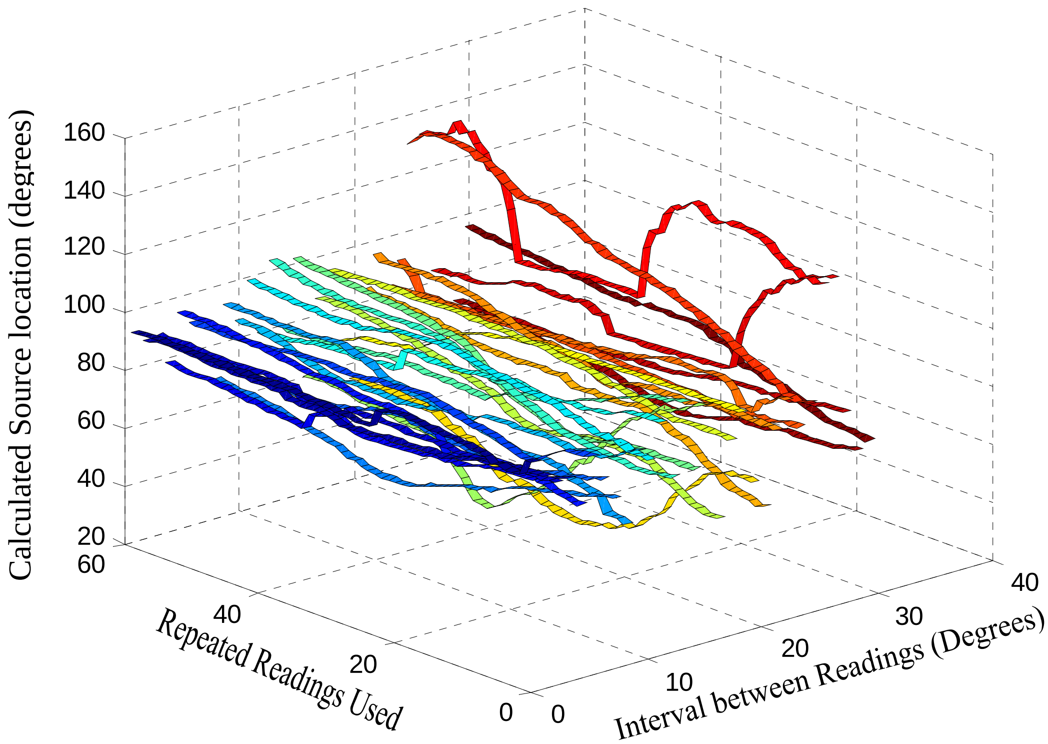

Figure 3-13. Plot of source location versus degree interval and repeats, ribbons interval.

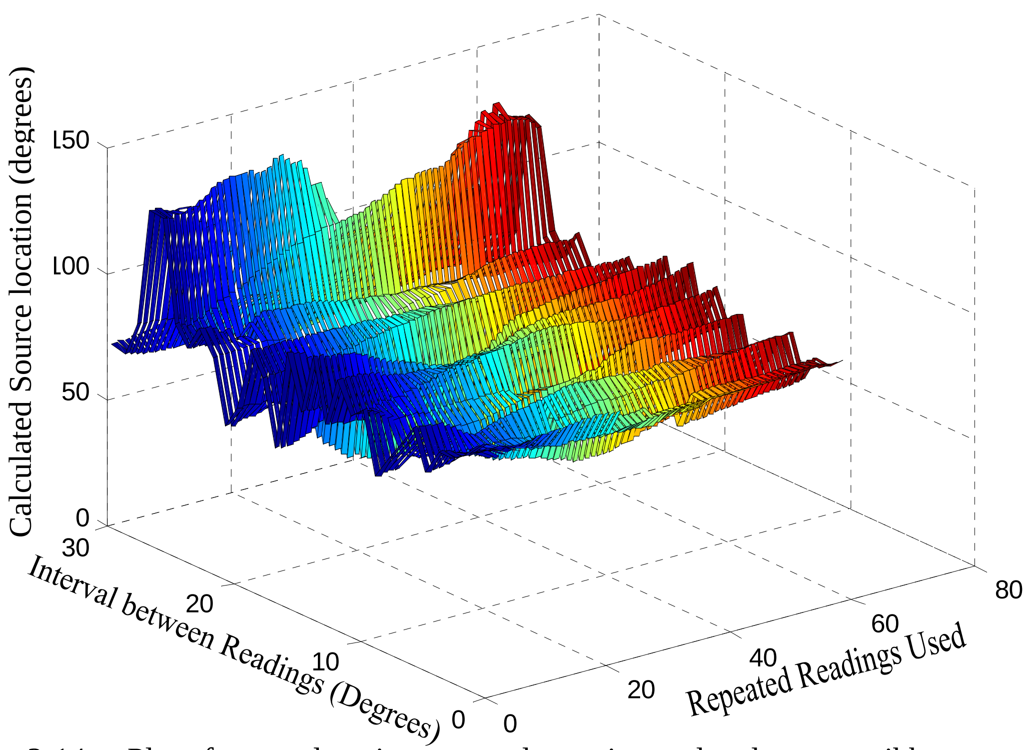

Figure 3-14. Plot of source location versus degree interval and repeats, ribbons repeats.

3.2 CONCLUSIONS

Based on the conclusions of the previous section, for the final test, a 15 degree interval between readings was chosen with 4 repeat readings, taking 8 seconds at each bearing for a total reading time of 96 seconds on readings, plus some time to turn. Once this final program was implemented it was found that, when Q > 5 for multiple readings, the robot is able to calculate the location of the source to within the error bar of the LIDAR bearing calculation. Figure 3-15 is a plot of the readings taken in a 180 degree search for a source at 90 degrees (again, shown in a thick green line, with the LIDAR error bar around it in thinner green). As can be seen in Figure 3-15, the source location is within the error bound of the LIDAR.

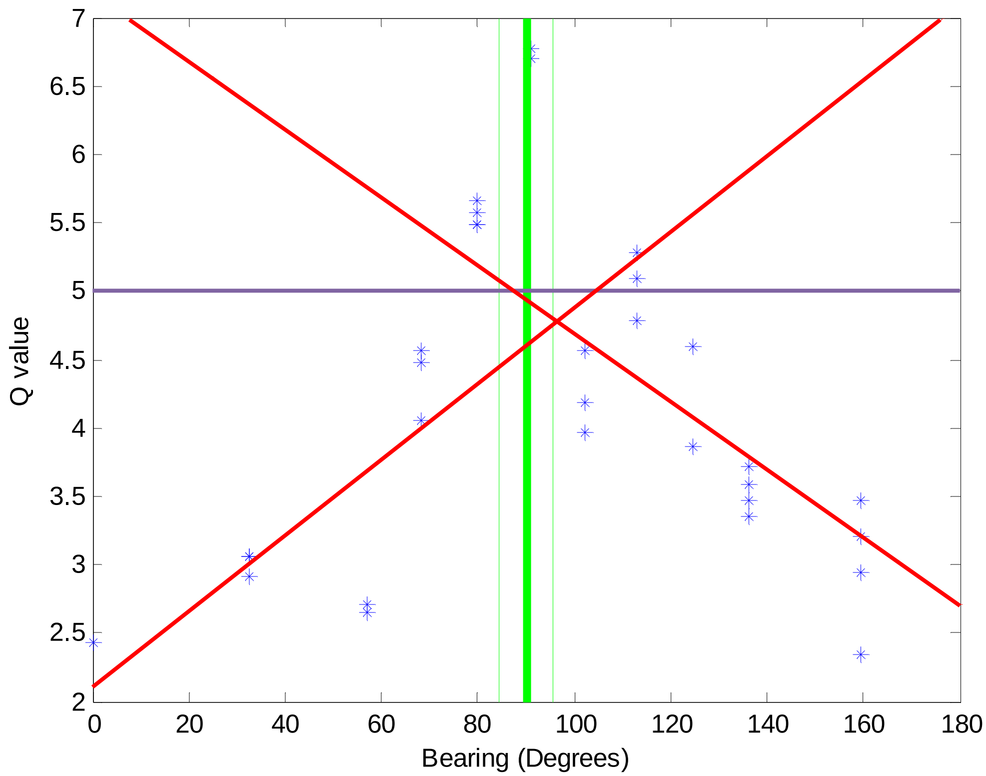

Figure 3-15. System response, with fitting and position estimate when Q>5 (near source).

Since few points exist on the graph, a curve fit would not have been possible using a 4th order polynomial fit. It is only the double line which allows the user to need so few data points to obtain accurate source location or bearing.

However, when Q<5 (near source) for most of the readings, then the robotic system produces result as shown in Figure 3-16. As shown in this figure, the location of the source is much less accurate compared to that found in Figure 3-15. Also notice how few readings there are with a Q value above 5. Thus it was concluded that the larger the number of readings with a Q value over 5 is, the more accurate the calculated bearing of the source.

Figure 3-16 shows a particularly bad situation, when the Q value detected by the IDM was so low that only one reading had a Q value above 5. Because all of the data points are so low, the data is extremely noisy. When the regression was calculated on the right side, the data appeared to be sloping towards the origin. This led to a physically impossible calculated source location.

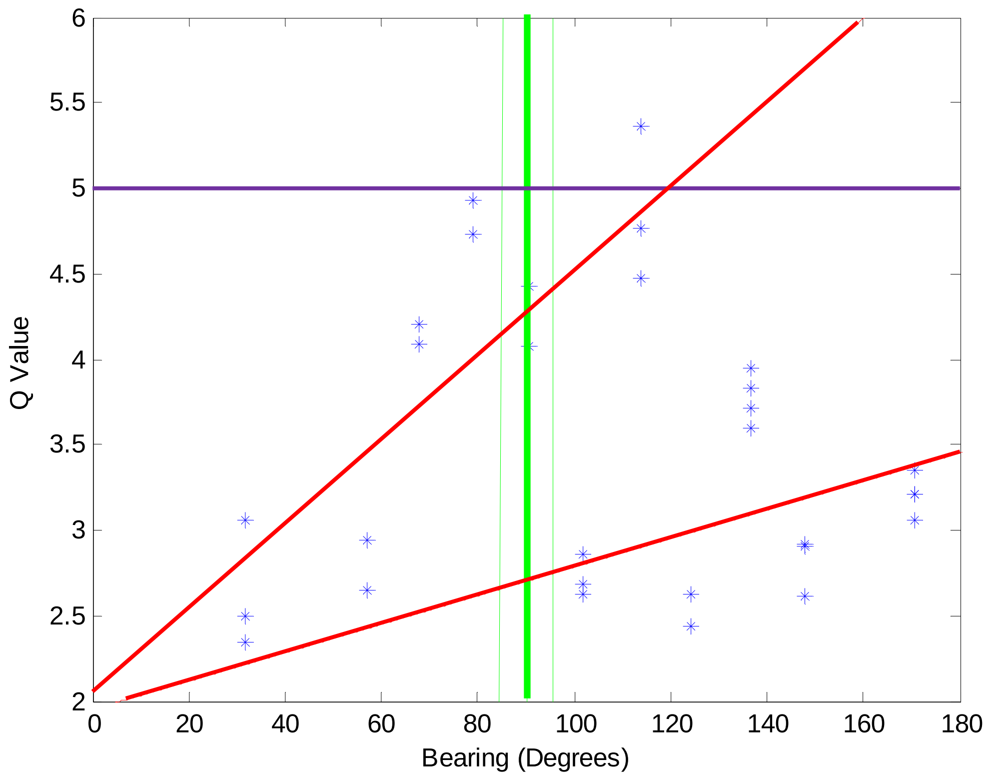

Figure 3-16. System response, with fitting and position estimate when Q<5 (near source).

It should be noted that even in the best of circumstances it is still necessary for the source to be at least 90 degrees away from the initial start point. However, this is due to time constraints in the creation of the final MATLAB program, and not to any physical limitations. A program able to account for the source being close to the beginning or end of the rotation, while not trivial, is certainly possible. There must also only be one of each source, because the current curve fit has no ability to resolve two overlapping signals. As before, this is not an insoluble problem just one that could not be solved in this time frame.

The main conclusion is that the presence of high Q values, greater than or equal to 5 provides an environment in which the robotic system can both identify the source and determine its location. High Q values can be obtained by searching for isotopes with high activities or when short distances exist between the isotope and the system's detector. However, the exact relationship between the Q value and an isotope's radioactivity was beyond scope of this project.

3.3 RECOMMENDATIONS

This robot is only a proof of concept, to show that the idea is feasible and worth further study. Hopefully there will be later generations of the product. There are some improvements that would benefit a robot professionally manufactured from basic items that were not available for this version, and these will be discussed in section 3.3.2, If another midshipman were to continue this project, there are some modifications to the existing frame and directions of research I would recommend, and these are discussed in 3.3.

3.3.1 Adjustments to the existing frame and directions of research

It was not possible to get the LIDAR range finder as accurate as hoped for at calculating the bearing of the robot. A digital compass, while inaccurate as the robot moves around the room and the deviation caused by nearby large metallic objects changes, is far more accurate at relative change as the robot rotates on the spot than the LIDAR has proved to be.

The next recommendation would be to rewrite the software which calculates where the source is so that it is more robust. Though it would not be trivial, it is possible to have the program set up so that it could handle, or at least identify the possibility, of multiple identical sources, or sources which start the reading within the IDM's field of view. At the moment, the software can only function under idealized lab conditions. The changes recommended above would allow the robot to function under more realistic conditions.

Though the ability to tilt the IDM is included in the construction with a linear actuator, it was not completed. The actuator needs power and a controller, and some form of angular measurement before it can be used. These items can be easily added if a future project decides to extend the current one to add a 3D search capability.

Perhaps it would be possible, with sufficient time, to program the robot do a quick turn to determine the strength of nearby sources. From that it would calculate a time window which would hopefully lead to enough Q values greater than 5 to allow the source to be located efficiently for that day. By having the robot automatically compensate for changing conditions, the unreliability of the source location could be reduced. Another possibility is to re-write the linear regression software to give more weight to larger Q values to reduce the effect of noise from the lower ones.

The IDM software can determine the presence of 71 isotopes or materials. However, in this study only the response of the system to Cs-137 was studied. Though a sample reading using Co-60 was taken, and appeared similar to Cs-137, this isotope was not examined in detail Another recommendation would be to study the system's response when isotopes other than Cs-137 are present to determine either if the relationships found in this study are still valid or whether the system will start producing false positives.

Finally, it would be nice for the robot to one day become completely autonomous, but there are clearly many steps between where the robot is now and that stage. However, this project has helped to shape the first stages, and provided a rough road map for the rest of the solution.

3.3.2 Recommendations for future manufacture

From experience gained troubleshooting, it would be desirable to avoid AC power altogether. Somewhere within the systems, the IDM uses both a 12V and a 24V DC power supply. All other items also use DC power, though the LIDAR draws it as AC and then converts it to 24V DC (details to follow). For this project it was decided that it was too risky to disassemble the IDM to get to these power supplies, due to the high value of the sensor. However, if this robot was to be taken to full production, it would be possible to work with ORTEC experts to directly power the system with batteries. It is estimated that the whole system could run from two or three batteries of different voltages, from the Rabbit board to the LIDAR to the IDM. Using DC to power everything would reduce complications and remove a major weakness of the power system. It would also increase the runtime the machine could achieve for the same weight of battery, by being dramatically more efficient with the power. While cooling down the system, it should be possible to supply DC power in parallel to the batteries, allowing for a two switch change from wall power (through a rectifier) with the batteries charging, to the batteries supplying the power.

On the programming side, everything currently done in MATLAB should be converted to C. Though MATLAB is far easier to use in the prototyping stage, it is dramatically less efficient to use in the final product than C. Fortunately, MATLAB was designed to make it easy to convert to C code if necessary. There are some matrix handling abilities of MATLAB that will be missed, but they can be replicated with subroutines. MATLAB is a wonderful developmental tool, but the program should be in C in the final product due to its greater speed. However, it would still probably be useful to have the locomotion run on a separate circuit to the rest of the program, as it requires constant low level attention.

As it stands, the most taxing terrain this robot can negotiate is a wheelchair ramp, and that is with a slight manual ‘bump’ up the first few inches. In many of the possible real-world scenarios, it would be desirable for the system to far more agile. However, if this proof-of-concept had been designed to be able to climb over rubble and up stairs, it would have taken far more time than the Naval Academy’s Trident scholar program permits to design the base. The only word of caution on this matter for future designers is rotating the nuclear sensor. Either the robot must have a zero turning radius or the HPGe sensor being used must be able to rotate independently of the base. Without this ability, it is not possible to detect the direction of a source and thus calculate its location.

SECTION 4

REFERENCES

Anzelon, G., Hammond, W., Nicholas, M., *"IAEA Illicit Trafficking Database Programme"*, IAEA Bulletin, issue 43, p. 15 April, 2001.

Bunn, G., Bunn, M., "*Nuclear Theft and Sabotage*", IAEA Bulletin, issue 43, p. 20-29, April, 2001.

Delikat, K., private communication, 2009

FEMA, "*Are You Ready? Radiological Dispersion Device*", http://www.fema.gov/areyoureadt/radiological_dispersion_device.shtm, 2006

Harper, M., Nelson, M., Lyne P., Bellomo, D., Camde, A., Schmitt, J., "*Technical Assessment of the ARDIMS on Airborne Platforms Under a Variety of Different Conditions*", Defense Threat Reduction Agency report, September, 2007

Knoll, G.F, "*Radiation Detection and Measurement*", 3rd edition, John Wiley and Sons, 2000.

Keyser , R. M., Twomey, T. R., *"Performance of a Small Portal Using Integrated Germanium Detector Modules"* INMM 49th annual meeting, Nashville, Tennessee, July 2008

Keyser , R. M., Twomey, T. R., *"False Positive Probability as a Function of Background for Short Data Collection Times in a Germanium Detector Portal Monitor"* INMM 49th annual meeting, Nashville, Tennessee, July 2008

Nelson, M.E., et al., "*Investigation into the Feasibility of Highly Enriched Uranium Detection by External Neutron Stimulation (Expanded Study)*", Defense Threat Reduction Agency report, May, 2006

OSHA, "*Safety and Health Topics: Radiological Dispersion Devices (RDD)/Dirty Bombs*", http://www.osha.gov/SLTC/emergencypreparedness/rdd_tech.html, 2004

ORTEC *"Technical specifications for Detective-ASP Series HPGe Interchangeable Detector Module (IDM)"*, www.ortec-online.com, 2009

Shea, D., "*Radiological Dispersal Devices: Select Issues in Consequence Management*", Library of Congress, Washington DC, 2004.

SICK AG, *"Technical specifications of the LMS200 indoor LIDAR"*, www.sick.com, 2009

APPENDIX A

LIST OF DETECTABLE ISOTOPES

This is the list is of isotopes which the IDM can identify in the current configuration. This was developed by Lawerence Livermore National Laboratory and ORTEC

Am241	Na22
Ba133	Neutrons & H present
Bi207	Np237
Cf252/Cf249	Pd103
Co56	Possible nuclear material
Co57	Possibly Cd109
Co60	Possibly weapon
Cr51	Plutonium
Cs131	Ra226
Cs134	Sb125
Cs137	Se75
Uranium	Sm153
Eu152	Sn113
Eu154	Sr89
Ga67	Tc99M
Ho166m	Th K x-rays
I123	Th232
I125	Tl200
I131	Tl201
In111	U232/Th228
Ir192	U233
K40	U235
Medical - positron emitter	U238
Mn54	Xe133
Mo99	Xe135
NORM background	Y88

APPENDIX B

IDM ROBOT OPERATING INSTRUCTIONS

I. Safety Precautions

The following safety precautions shall be closely adhered to in order to minimize risk of equipment damage and/ or personal injury

1. Ensure unit is plugged into AC power source prior to energizing components
2. Do not restart IDM while operating on battery power
3. To prevent electrical shock, prior to performing any work on systems, disconnect AC power and remove fuses from batteries.

II. Prestart Checks

The following procedures shall be performed prior to disconnecting from AC power

1. Ensure batteries are fully charged
2. Ensure IDM has been cooling for a minimum of 4 hours and wattage reading is, at its highest, below 90 Watts.

III. System Initialization

The following procedures shall be performed when initializing the LIDAR

1. Ensure that the power supply for the LIDAR is plugged in.
2. Ensure that the LEDs on the front of the LIDAR display a green light, rather than a red and/or orange light.
3. Ensure that the cable from the LIDAR to the serial to USB cable is connected.
4. Ensure that the Serial to USB cable is connected to the laptop.
5. Ensure that the ES308 Rabbit board is switched on, with one red LED flashing.
6. Ensure that the cable from the Rabbit board to the Serial to USB cable is connected.
7. Ensure that the cable from the serial to USB cable is connected to the Laptop.

The following procedures shall be performed when initializing the software

1. Ensure that the laptop is switched on.
2. Logon to the “Nuke” user. Do not use the ‘Cadig’ user.
3. Allow system to completely initialize before starting any software.
4. Ensure that the data acquisition program has initialized. If it has not, manually start it using the shortcut on the desk top.
5. Ensure that the display of the data acquisition system looks like Figure B-1, with the highlighted column in shades of blue and green, rather than Figure B-2, with the highlighted column looking identical to neighboring columns. If it looks like Figure B-2, see trouble shooting section for IDM to laptop communications.
6. Check that the “Current Temperature” reads approximately 90K. If it does not, allow IDM further time to cool.
7. Check that the “low CRM” reads “No”. If it does not, allow IDM further time to cool.
8. Initiate MATLAB.
9. Open code to be run, for example “SourceFinder” or “BookkeeperWindow”.
10. Open “Shortcut to IDM Demo Program”. This should open in Notepad.

11. Ensure that "time window" in the code to be run matches that of the IDM demo program startup instructions.
12. Close "Shortcut to IDM Demo Program".
13. Initialize "IDM Demo Program".
14. Set to "Search Mode"
15. Wait for duration of time window and ensure that the IDM demo program is able to detect sources.

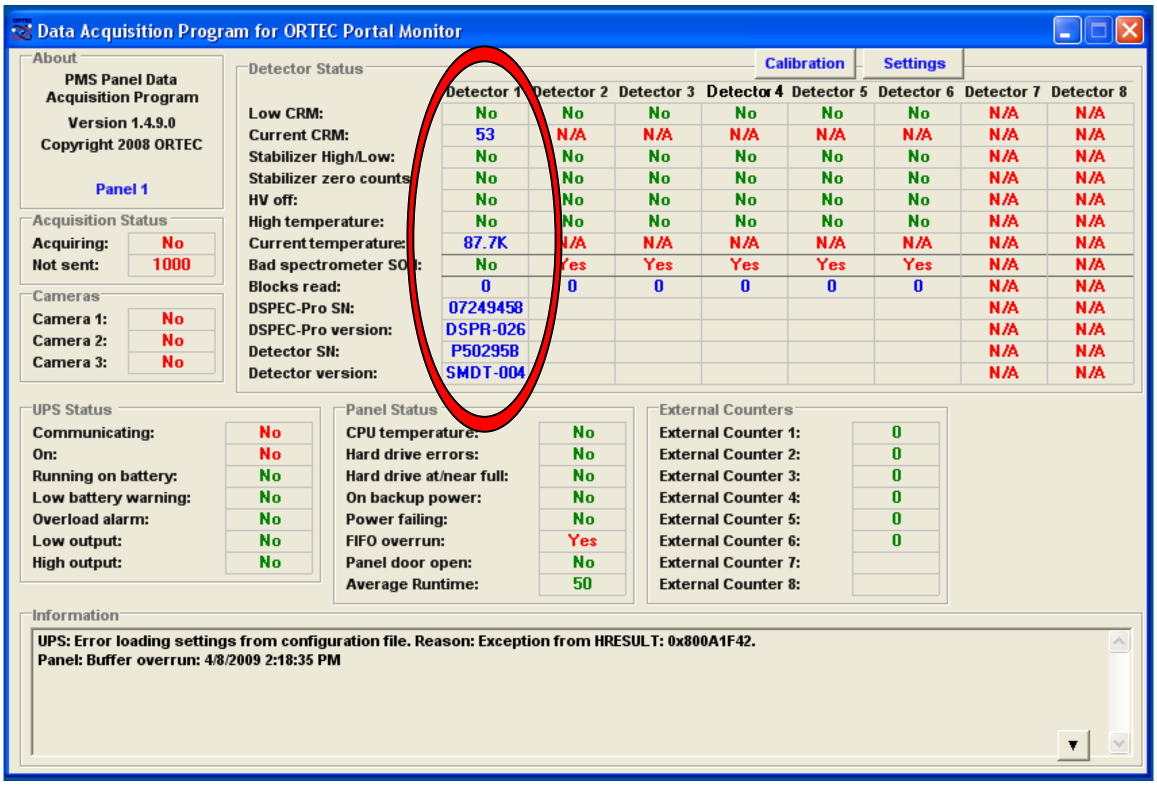

Figure B-1. Image of Data Acquisition program when IDM connected

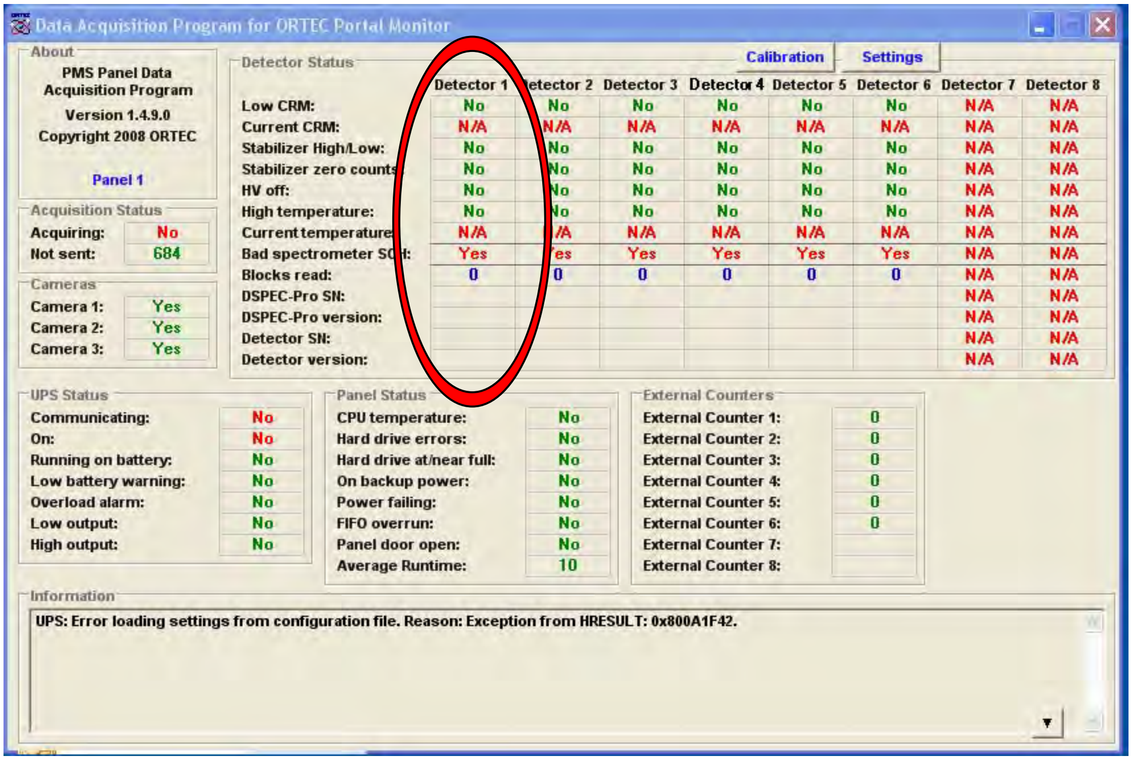

Figure B-2. Image of Data Acquisition program when IDM disconnected

IV. Movement Control

The following procedures shall be performed when manually moving robot

1. Ensure that ES308 Rabbit board is switched on with one red LED flashing.
2. Ensure that the cable linking the Rabbit board to the Laptop is connected.
3. Define the serial connection within MATLAB with the following code:
   ```
   s = serial('COM5', 'BaudRate', 14400);
   ```
4. Open the Serial Communication within MATLAB with the following code:
   ```
   fopen(s)
   ```
5. To initialize one meter of forward motion, use the following code:
   ```
   fprintf(s, 'f1x')
   ```
6. The above code can also be used to make the robot go back a desired distance, buy inserting a negative sign before the distance.
7. To initialize a rotation of 10 degrees to the left, use the following code:
   ```
   fprintf(s, 't10x')
   ```
8. The above code can be used to make the robot rotate to the right by inserting a negative sign before the distance.
9. To initialize a rotation at 1 degree a second to the left, use the following code,
   ```
   fprintf(s, 'c1x')
   ```
10. To change the speed of the rotation, send the above code again with a different rotation rate. To rotate to the right, insert a negative sign.
11. To cease rotation, use the following code. It may be necessary to send it multiple times.
    ```
    fprintf(s, 'sx')
    ```

12. When finished using manual motion control, close serial connection using the following code:
```
fclose(s)
```

The following procedures shall be performed when using RC control of the robot

1. Ensure that ES308 Rabbit board is switched on with one red LED flashing.
2. Ensure that the cable linking the Rabbit board to the laptop is connected.
3. Ensure that the Xbe transceiver is switched on. The location of the power switch is shown in Figure B-3. The red LED should flash. If it does not, replace the battery and try again.
4. Ensure that the joystick is switched on. The location of the power switch is shown in Figure B-3. The red LED should be on solidly. If it does not, replace the battery and try again.
5. Ensure that the gain wheel on the left side of the joystick, highlighted in Figure B-3, is rotated all the way to the top.
6. Define the serial connection within MATLAB with the following code:
```
s = serial(‘COM5’, ‘BaudRate’, 14400);
```
7. Open the Serial Communication within MATLAB with the following code:
```
fopen(s)
```
8. Initiate Radio Control by using the following code
```
fprintf(s, “r2x”)
```
9. Use four directional button, highlighted in Figure B-3, to move robot. Push button forward to make the robot go forward, push button to the right to rotate on the spot to the right and so on.
10. It can be necessary to send the above code multiple times before the robot responds to the joystick. Always ensure that the gain wheel is rotated up.
11. When RC control is complete, roll gain switch down.
12. Ensure that joystick and Xbe are switched off, with no red LEDs showing.
13. Use the following code to close the serial connection:
```
flcose(s)
```

The following procedures shall be performed when manually interacting with LIDAR

1. Ensure that LIDAR LED array is showing a solid green light.
2. Ensure that cable linking LIDAR and Laptop is connected.
3. Define the serial connection within MATLAB with the following code:
```
s1 = serial(‘COM7’, ‘InputBufferSize’, 1024);
```
4. Open the serial communication within MATLAB with the following code:
```
fopen(s1)
```
5. Ensure that in any use of the LIDAR by a subroutine, the ‘s1’ serial connection is passed to the program.
6. Once manual work is finished, close serial port before using circular search program using the following code:
```
fclose(s1)
```

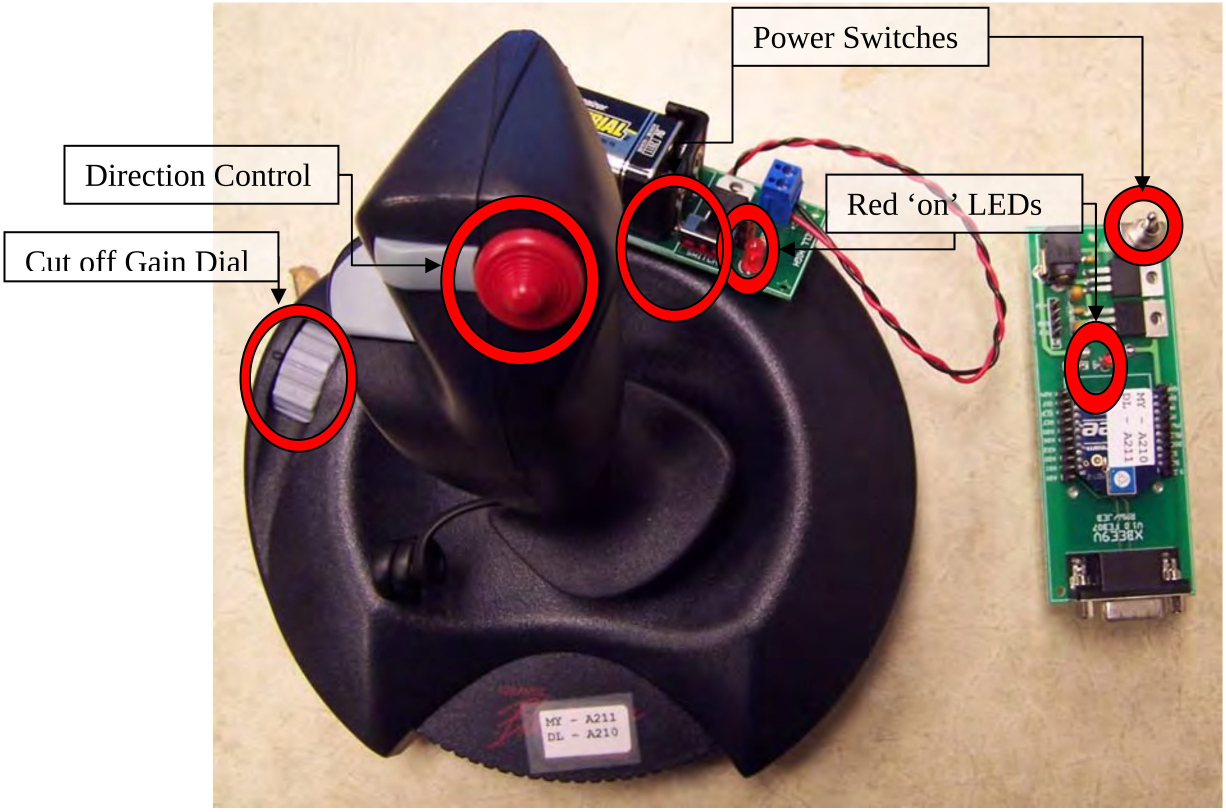

Figure B-3. Picture of the RC joystick and the Xbe transceiver.

V. Operation of Circular Search

The following procedures shall be performed when calculating the location of the source

1. Ensure that all serial communication lines, both 's' used for the Rabbit board and 's1' used for the LIDAR, are closed using the following lines of code:
   ```
   fclose(s)
   fclose(s1)
   ```
2. Open up the Matlab script "SourceFinder"
3. Bring up the IDM Demo Program search screen.
4. Ensure that data is being collected and sources detected.
5. Double check that the window length shown in the code matches with the start and end times of the identifications shown on the IDM demo program.
6. Ensure that the source is not closer than 90 degrees to the start position of the robot.
7. Ensure that there will not be two identical sources detected during the rotation.
8. Run "SourceFinder" script.
9. The current dead reckoning bearing should be printed on the screen following each rotation.
10. There should be 7 cases of "Message Correct = 3, Count = 728' printed on the screen between each dead reckoning bearing.
11. If a long string with a "Message Correct" equal to 1 or 2 is printed, use ctr-c to end the program, restart the LIDAR and restart this process from step 1.

12. Once the rotation is complete, a plot with two subplots will appear, comparing the LIDAR ranges to calculate the bearing of the robot at each set of readings.
13. A list of isotopes will be displayed with a number next to each. Type in the number next to the desired isotope. Ensure that it is an integer.
14. A chart displaying the data and the curve fit will appear, and the source location will be printed in MATLAB. All variables will still be available for further analysis.

VI. Two Dimensional Location

The following procedures shall be performed when calculating the location of the source

1. Ensure that the X and Y co-ordinate location of the robot in meters and the bearing of the source from the Y axis is recorded for three locations
2. Use the following code to calculate the location of the source.
   ```
   [XCoordinate, YCoordinate, Range, Bearing] = XYcalc(x1, y1,
   b1, x2, y2, b2, x3, y3, b3)
   ```
3. X will be the x coordinate of the source, Y will be the y coordinate of the source.
4. Range will be the range of the source from the third position
5. Bearing will be the bearing of the source from the third position relative to the Y axis.
6. IFORGET will be a measure of the precision of the estimated location of the source. The smaller it is, the more precise the triangulation was.
7. A chart will appear, displaying the positions, lines of position and the estimated location of the source.

VII. Securing system

The following procedures shall be performed when closing down the system for charging

1. Ensure that Xbe and joystick are switched off.
2. Ensure that UPS extension cable is plugged into wall power.
3. If using an extension cord, ensure that the extension cord is switched on.
4. Ensure that the laptop is switched off and recharging batteries.
5. Ensure that IDM is on and cooling down.

The following procedures shall be performed during an emergency shutdown situation

1. If the IDM must be shut down for an emergency or restarted, remove the black power cord from the back of the machine.
2. If the LIDAR must be shut down for an emergency or restarted, remove the serial connector with only the black and red wires feeding into it.
3. If the ES308 Rabbit Board must be shut down for an emergency or restarted, switch off the board or, if this is insufficient, remove its power cable. This should halt any motion by the robot and restart the Rabbit's input.
4. In case of an electrical fire or other such emergency while the system is running on battery power, remove the fuses from the batteries. Then proceed to extinguish any remaining fires.
5. In case of an electrical fire or other such emergency while the system is running on wall power, unplug the power and remove the fuses from the batteries.

APPENDIX C

TROUBLESHOOTING AND DEVELOPMENT GUIDE

This section will focus less on the theory of what is happening and more on practical instruction on how to avoid mistakes in the system. Certain sections of the robot, like the power supply and the serial communication with MATLAB, are very delicate and can easily fail. This section's goal is the pass on those of the practical "dos and don'ts" learned the hard way while developing this system which are not detailed in Appendix B, as well as hints for future developers using the system. This section also brings out the array of factors that must be properly aligned for the system to successfully collect data.

C.1 INITIALIZING SYSTEM

This machine is first-of-a-kind proof of concept. It is not a fully integrated system; it is a patch work of independent parts which interact with each other. It is therefore extremely important to make sure that everything is initialized properly, to interact with everything else. These instructions assume that the whole system is completely de-powered with nothing on at initialization.

C.1.1 Powering up system

This is possibly the most delicate stage in using this machine. If a mistake is made in any stage after this the most you will likely have to do is restart the computer or plug in the machine to restart the IDM. If a mistake is made with the power supply, you will likely have to get out a soldering iron and some replacement parts.

As mentioned in Appendix B, if it is necessary to restart the IDM, make sure that the system is plugged into the wall even if it is cooled down. This is because the initial power spike of a cooled down IDM has taken out the diode in the past (see Figure 2-5), requiring a whole section of wiring to be redone. The diode theoretically has a much higher maximum power and voltage, but it is safer to plug in the system when restarting the IDM.

C.1.2 Initializing required software and communications

Two items of software are needed for any readings from the IDM. The first is the Data Acquisition Program, which manages the IDM and displays its status. Though it doesn't actually calculate the Q value, it manages communications with the IDM and is essential to data collection. This program should initialize on start up. If necessary, it can be started from the icon. If the Data Acquisition Program (DAP) is able to connect to the IDM, its display will look something like Figure C-1. Notice the circled rows of blue figures. The DAP was modified for this application. Initially the DAP was designed to coordinate the results of 8 different IDMs which is why there are 8 detector columns in Figure C-1. This feature will allow the addition of multiple IDMs in any extension project. If the IDM does not interface with the laptop it should look as shown Figure C-2. As shown in Figure C-1 the row for Detector 1, circled in red, looks identical to all of the others which are not connected.

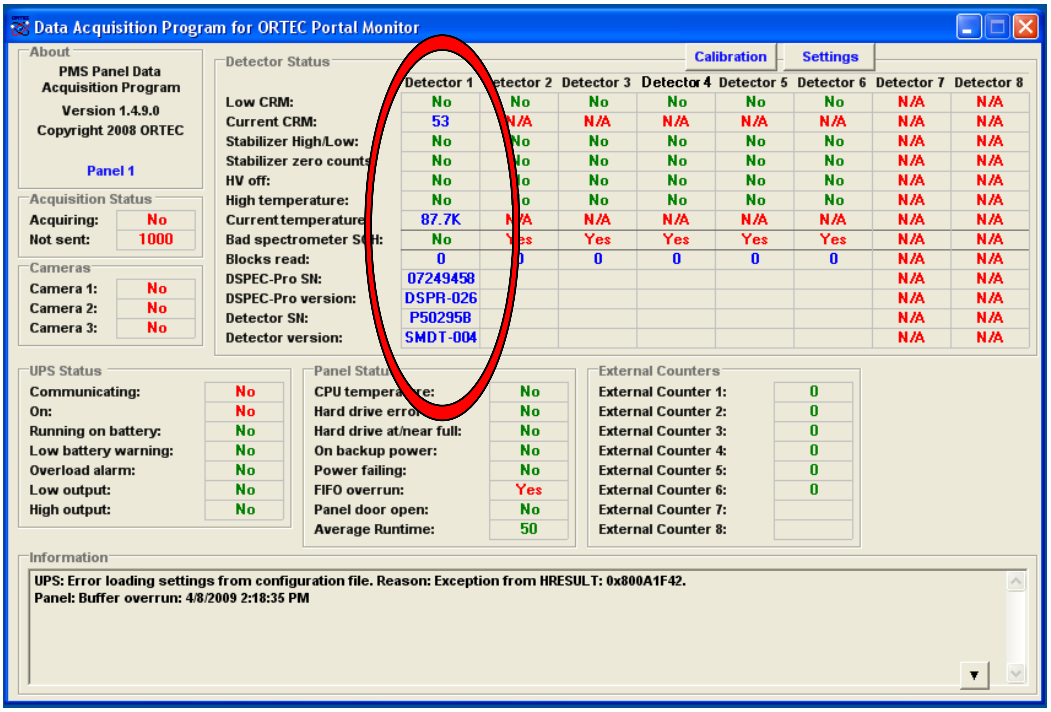

Figure C-1. Image of Data Acquisition program when IDM connected

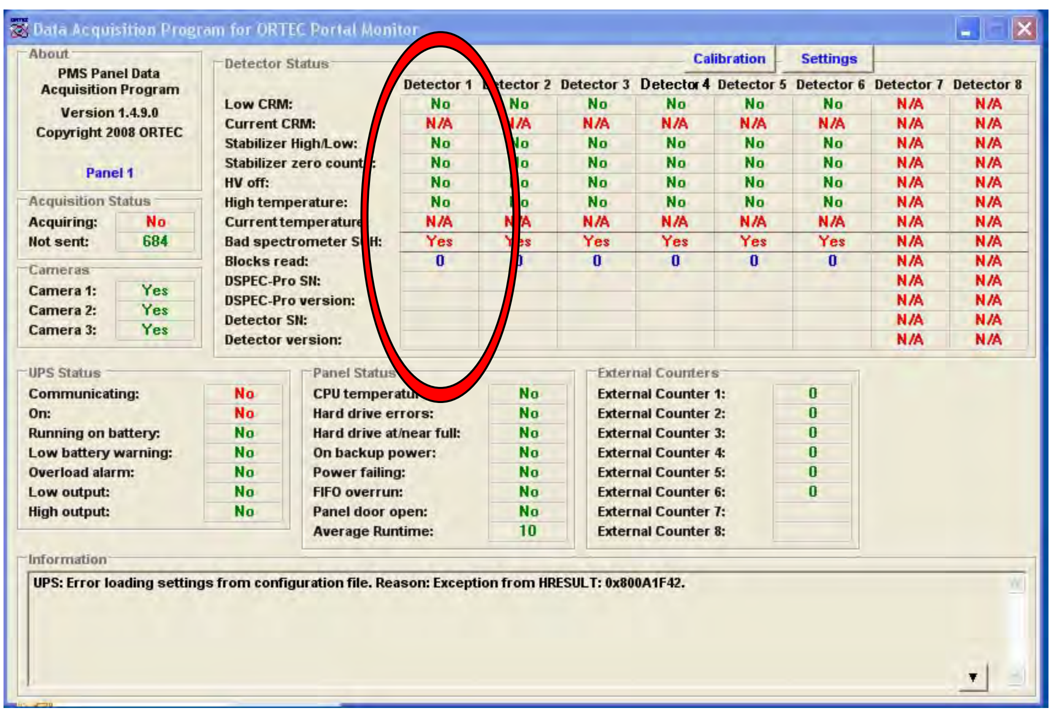

Figure C-2. Image of Data Acquisition program when IDM disconnected

If this should happen, try restarting the laptop computer and/or restarting the IDM (remembering to plug the system in if restarting the IDM). If neither of these actions is effective, the final method is to switch off both systems (i.e. computer and IDM), leave them to unwind for 5 minutes, and then restart them both. This clears the memory of the USB server, and thus reset the communication between the two pieces of hardware. This final debug method has yet to fail, short of a major malfunction with the IDM.

The second most important software is the IDM Demo Program beside the MATLAB program. First and foremost, you must check that the window length shown in the initialization code matches the one being used in the MATLAB program. A short cut to this code is on the Laptop's Desktop. This program must be restarted before any changes to this code take effect. This software must then be initialized and set to 'Search' mode. If this step is not completed then no data will be passed to MATLAB, which will be interpreted as a constant stream of negative results. It should be noted that if you change the mode, the system will invariably shutdown. In this case, restart the program, and set to "Search Mode" immediately. The "Automatic" mode starts detecting the sources whenever a source is detected above the threshold by the IDM. The "Manual" mode continually collects data, using every piece of data collected to calculate the Q value. These other two modes will not work with the MATLAB script "SourceFinder".

Once the IDM software is initialized, then MATLAB can begin. The system for interfacing serial communication and MATLAB is far from ideal, but serial communication is vitally important. Serial communication controls both the wheels and the LIDAR. Without serial communication, the machine is paralyzed and blind. Any serial communication with the wheels must first be defined and then opened, as shown below:

```
s = serial('COM5', 'BaudRate', 14400);
fopen(s)
```

For the LIDAR, it must be as shown below:

```
s1 = serial('COM7', 'InputBufferSize', 1024);
fopen(s1)
```

The baud rate defined must match that in use by the LIDAR and the Rabbit board. If the baud rates do not agree, communication is impossible. In the second case, if the incoming buffer is not expanded from the standard 512 to 1024, there is insufficient space for all of the data which the LIDAR outputs. This is the difference between receiving 135 degrees of data with no check sum and receiving 180 degrees of data with the check sum. On a final note, COM5 and COM7 are the serial ports to which the rabbit and the LIDAR are connected in this setup. If the setup is changed or remade from scratch, ensure that the serial ports correctly correspond to each of these external sources.

It is important to close these communications if they are used in the command line before running the program. The program assumes they have not been run and will try to open the serial ports. If they are already open, it creates and error and is unable to use the port again. The only way to cure this is to restart MATLAB. The first troubleshooting step in any issue with serial communication should be to restart MATLAB. If you wish to avoid that, it is necessary to type either “fclose(s)” or “fclose(s1)” before you try to run the program. Once you get a red error message about the serial ports, there is nothing short of restarting MATLAB which will help you. If you can close the serial port before you start the program and trigger the error message, it will all run smoothly.

Note that within the MATLAB programs, the serial ports are passed from routine to subroutine. Therefore, if calling any of the programs which use serial communications in isolation, it is necessary to define and open the serial port in the command line and then pass it into the subroutine.

3.2 BASIC SYSTEM CONTROL

Though there were plans to create a more user friendly system, time ran out. Instead, the next user is left with the skeleton communication system created with the Rabbit board which was developed for use during the research. However, this does make it easier to use the connection in any subsequent programs.

3.2.1 Practical motion control & RC

When the Rabbit motor control was created, four modes were created. A mode for linear travel, a mode for rotational travel, a mode for variable speed rotation and a radio control mode. The details of how to call these programs are given in Appendix B. The following is an example of the code needed to make the robot go forward one meter.

```
fprintf(s, ‘f1x’)
```

To break this down into sections, ‘fprint’ is a MATLAB function which outputs the string (surrounded by quotation marks) to which ever output follows. Normally, a ‘1’ is used to print the results to the main MATLAB command screen. However, to communicate with the rabbit, the ‘s’ refers to the Serial port, which was hopefully opened previous to the attempt to make the robot move. The ‘f’ tells the rabbit to move in a line, and the ‘1’ tells it how many meters. If there is a negative sign in front of the meters, then the robot will move backwards. The ‘x’ at the end informs the rabbit that the message is over, to facilitate sending messages of multiple lengths.

For the turn mode there are two options. The first is the one used within the circular search program: a subroutine called ‘turn’. The following is an example of how to call this subroutine to make the robot turn 10 degrees clockwise.

```
turn(10, s)
```

As mentioned before, because of how MATLAB handles serial ports, it is necessary to pass the already opened serial port into the function. The '10' is a request to turn 10 degrees clockwise. Turn takes advantage of the '1' send back down the serial connection when the instruction is received and the '2' sent when the action is complete to ensure that the instruction is carried out without error. The use of this feature is strongly recommended in any future system which uses the existing rabbit controls to create an autonomous system. Once the turn function believes that the action is complete, it will return a 1 within MATLAB. Until then, it returns a 0. This means that if you want to wait until the turn is complete, all you have to do is use:

```
while(not(turn(10,s)));end;
```

This will pause the program at that point until the turn is complete. The second option is to access the instruction directly as was done above with linear motion. The third mode which the rabbit offers is the circular rotation mode. Both of these are detailed in Appendix A.

One final note on use of the radio control feature. Only a few of the buttons on the Joystick, as shown in Figure C-3, are actually used. The gain wheel on the side of the joystick is used as the cut off signal. Make sure that this is rotated up as far as it will go when you have both the Joystick and the Xbee turned on with the 'r2x' signal sent. If it is not, then the rabbit will get the cut off signal, and the robot will not move. Often the cut off is still activated within the internal memory of the rabbit, so it is necessary to send the 'r2x' signal more than once. It is recommended to send the signal, try to move the robot and if it doesn't respond, check the LED's, the cut off gain dial and then resend the signal.

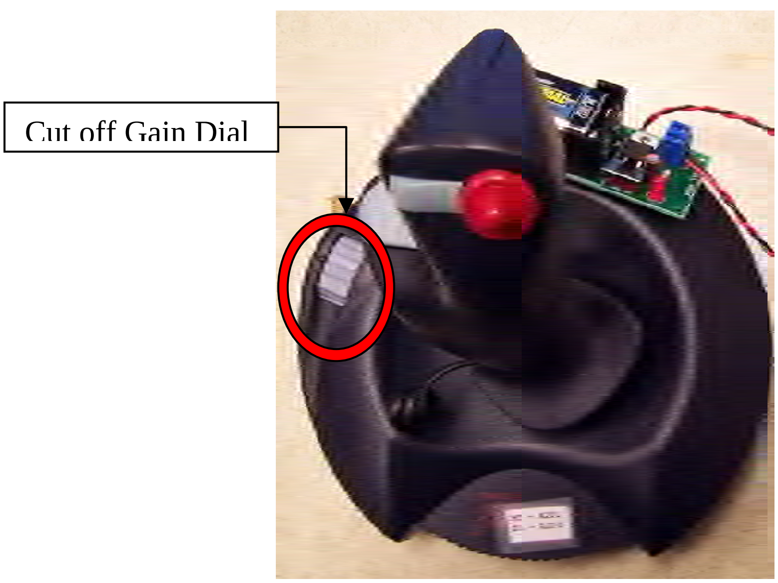

Figure C-3. Picture of the RC joystick